TECHNISCHE UNIVERSITÄT
CHEMNITZ

FAKULTÄT FÜR INFORMATIK

Diplomarbeit

Analyse und Vergleich von Extraktionsalgorithmen für die Automatische Textzusammenfassung

von

Monique Krübel

Matrikelnummer: 25423

Betreut von: Dr. Maximilian Eibl

Erklärung

Hiermit versichere ich, dass ich die Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Chemnitz, 06.04.06

Inhaltsverzeichnis

1. Einleitung

Obwohl schon seit den 50er Jahren auf dem Gebiet der Automatischen Textzusammenfassung Forschung betrieben wird, wurden der Nutzen und die Notwendigkeit dieser Systeme erst mit dem Boom des Internets richtig erkannt.

Das World Wide Web stellt eine rasant wachsende Menge an Informationen zu nahezu jedem beliebigen Thema zur Verfügung. Um den Zeitaufwand zum Finden und auch zum Wiederfinden der richtigen Informationen zu minimieren, traten Suchmaschinen ihren Siegeszug an. Doch um einen Überblick zu einem ausgewählten Thema zu erhalten, ist eine einfache Auflistung aller in Frage kommenden Seiten nicht mehr adäquat. Zusätzliche Mechanismen der Textaufbreitung wie beispielsweise Extraktionsalgorithmen für automatische Zusammenfassungen können hier helfen, Suchmaschinen oder Webkataloge zu optimieren.

Zusammenfassungen dezimieren den Zeitaufwand und versorgen den Nutzer schnell und hochinformativ mit den wichtigsten Fakten. Daher kann man Zusammenfassungen in jedem Bereich des täglichen Lebens antreffen. Egal ob es sich dabei um Nachrichten, Schlagzeilen, Filmvorschauen, Sitzungsprotokolle oder Inhaltsangaben von Büchern handelt, jede dieser Zusammenfassungen ist wesentlich kürzer als das Original und bietet einen Überblick über den wesentlichen Inhalt eines Textes, Filmes oder Ereignisses ohne den Inhalt dabei zu interpretieren oder zu bewerten.

Allerdings ist es heute meist noch Aufgabe von Autoren, Reportern oder professionellen Extraktern manuell zu entscheiden, welche Teile des Originals ausgelassen werden können und welche unbedingt in die Zusammenfassung übernommen werden müssen, so dass die Zusammenfassung trotz ihrer Kürze als Mittel zur Sacherschließung dienen kann. Die Automatisierung dieser Extraktionsentscheidung ist der Kernpunkt der Automatischen Textzusammenfassung.

In der Diplomarbeit soll eine Analyse von Extraktionsalgorithmen durchgeführt werden, welche für die automatische Textzusammenfassung genutzt werden können. Auf Basis der Analyse als viel versprechend eingestufte Algorithmen sollen in Java implementiert werden und die mit diesen Algorithmen erstellten Zusammenfassungen in einer Evaluation verglichen werden.

1.1. Gliederung der Arbeit

Im zweiten Kapitel soll der so genannte „State of the Art“ auf dem Gebiet der Automatischen Textzusammenfassung dargelegt werden. Dabei werden die unterschiedlichen Ansätze und Methoden bei der Erstellung von Zusammenfassungen vorgestellt. Des Weiteren wird in diesem Abschnitt die Entwicklung der Automatischen Textzusammenfassung von den Anfängen der Forschung in den 50er Jahren über die Renaissance seit den 90er Jahren bis hin zu aktuellen Forschungsprojekten betrachtet. Dabei werden auch die ersten wegweisenden Algorithmen von Luhn und Edmundson vorgestellt.

Das dritte Kapitel befasst sich mit unserer Implementierung verschiedener vorgestellter Extraktionsalgorithmen sowie einem eigenen Extraktionsalgorithmus in Java. Dabei werden die grafische Nutzeroberfläche, verwendete Klassen, das Einlesen des Originaltextes, die unterschiedlichen Methoden zur Satzauswahl und die Erstellung der endgültigen Zusammenfassung erläutert.

Im vierten Kapitel geht es um die Evaluierung von Textzusammenfassungen. Es werden dabei auftretende Probleme aufgezeigt und die Klassifizierung der Evaluierungsmethoden dargelegt. Hierbei werden auch Standardisierungsbestrebungen wie die Document Understanding Conference und die TIPSTER SUMMAC – Evaluation vorgestellt. Im letzten Teil dieses Kapitels wird die Evaluierung des von uns erstellten Zusammenfassungsprogramms thematisiert. In diesem Zusammenhang werden wir auf die Auswahl der Evaluierungsmethoden, die Auswahl der Evaluierungstexte sowie die Durchführung und die Ergebnisse der Evaluation eingehen.

Im fünften Kapitel werden die Ergebnisse der Evaluierung zusammengefasst und es werden Probleme unseres Zusammenfassungsprogramms geschildert. Außerdem werden einige Erweiterungs- und Verbesserungsvorschläge beschrieben.

Im sechsten und letzten Kapitel werden verschiedene Anwendungsmöglichkeiten für die Automatische Textzusammenfassung geschildert.

2. Stand der Technik bei der Automatischen Textzusammenfassung – Ansätze und Algorithmen

Die Automatische Textzusammenfassung, deren Anfänge bis in die 50er und 60er Jahre zurückreichen, hat seit den 90er Jahren enorm an Bedeutung gewonnen. Gründe dafür sind die verbesserte Rechentechnik aber vor allem das enorme Wachstum der Textmengen, gerade im Internet.

Ein Beispiel: Für die Suchbegriffe „automatic text summarization“ erhält man von Google™ 96800 Vorschläge für Internetseiten, die sich mehr oder weniger mit dieser Thematik befassen. Sie alle zu lesen ist undenkbar. Interessant wäre ein Programm, bei dem man die Größe der gewünschten Zusammenfassung all dieser Seiten einstellt und welches dann die wichtigsten Aussagen aus all diesen Seiten zu einer sinnvollen Zusammenfassung zusammenstellt.

2.1. Steuerungsparameter für die Zusammenfassungserstellung

Je nach Zweck und Ziel der Zusammenfassung müssen unterschiedliche Parameter für die Erstellung der Zusammenfassung berücksichtigt werden. Beispielsweise muss für die Zusammenfassung einer Internetseite, die bestimmte Suchworte enthält, bei Google ein anderes Verfahren angewendet werden als bei der Zusammenfassung einer mehrbändigen Buchreihe.

Um den Typ der Zusammenfassung zu spezifisieren müssen folgende fünf Fragen, welche im Anschluss noch genauer erklärt werden, geklärt werden:

- Soll ein Extract oder ein Abstract das Resultat der Zusammenfassung sein?
- Wird eine allgemeine oder eine nutzer-orientierte Zusammenfassung gewünscht?
- Soll die Zusammenfassung informativ oder indikativ sein?
- Ist ein kohärenter Text gewünscht oder reicht eine Sammlung von Exzerpten?
- Sollen ein oder mehrere Texte als Grundlage für die Zusammenfassung dienen?

Es gibt zwei verschiedene Arten von Resultaten, die bei der Textzusammenfassung entstehen können: ein Extract oder ein Abstract.

Ein *Abstract* oder Kurzreferat ist eine kurze Inhaltsangabe eines Textes. Dabei wird der zentrale Inhalt beziehungsweise die Kernaussage aus dem Text extrahiert und anschließend neu formuliert. Hierbei können auch nicht im Originaltext enthaltene Informationen einfließen. Ein Abstract hat meist zwischen 250 und 300 Wörtern. Eine Erstellung von

Abstracts ist nur sehr schwierig zu automatisieren, da Textverständnis, ein gewisses Maß an Hintergrundwissen sowie eine sehr gute Spracherzeugung benötigt werden.

Bei einem *Extract* werden wichtige Textfragmente unverändert übernommen und erscheinen in der Zusammenfassung abhängig ihrer Wichtigkeit und Reihenfolge im Originaltext. Ein Textfragment kann dabei ein Absatz, ein Satz, eine Wortgruppe oder auch nur ein einzelnes Wort sein. Eine Erhöhung des Kompressionsgrades führt dazu, dass mehr Sätze aus dem Originaltext verworfen werden und nicht in den Extract übernommen werden.

Sehr wichtig für den späteren Nutzen der Zusammenfassung ist die Festlegung, ob eine allgemeine oder eine nutzerorientierte Zusammenfassung entstehen soll. Eine *allgemeine* Zusammenfassung gibt alle thematischen Schwerpunkte des Originals wieder. Im Gegensatz dazu enthält eine *nutzerorientierte* Zusammenfassung nur die thematischen Schwerpunkte des Originals, welche mit den Interessen des Nutzers oder einer aktuellen Frage des Nutzers korrespondieren.

Bei der Entscheidung, ob eine indikative oder eine informative Zusammenfassung gewünscht wird, spielt es eine Rolle, ob nur eine Kategorisierung des Textes benötigt wird oder ob das Verstehen des Textes das Ziel ist. Für das Herausfiltern der Hauptinformationen und die Erstellung sehr kurzer Texte kann eine *indikative* Zusammenfassung genutzt werden. Bei einer *informativen* Zusammenfassung wird der Originaltext auf das Wichtige verkürzt, so dass die wesentlichen Aussagen und bestimmte Informationen wiedergegeben werden.

Meist wird unter einer Zusammenfassung ein zusammenhängender Text verstanden, obwohl auch stichpunktartige Exzerptsammlungen das Ergebnis einer vom Menschen erstellten Zusammenfassung sein können. Das Ergebnis maschineller Zusammenfassungen sind „meist Sequenzen von Exzerpten aus dem Quelltext, sogenannte Extrakte“ [Guendogan & Schimpfky 2005, S. 4].

Die Entscheidung, ob ein oder mehrere Texte betrachtet werden sollen, muss abhängig vom Zweck der Zusammenfassung getroffen werden. Will man wissen, was in einem bestimmten Text steht, so wird natürlich nur dieser Text für die Zusammenfassung herangezogen. Will man sich aber zu einem bestimmten Thema informieren oder eine bestimmte Frage geklärt haben, kann man mehrere Texte, die dieses Thema mehr oder weniger behandeln, als Eingabe für den Zusammenfassungsprozess nutzen.

2.2. Zwei unterschiedliche Ansätze zur Textzusammenfassung

Es existieren im Prinzip zwei komplementäre Ansätze zur Erstellung von Zusammenfassungen: die Textextraktion und die Faktenextraktion.

2.2.1. Faktenextraktion

Bei der Faktenextraktion oder Informationsextraktion, handelt es sich um einen geschlossenen Ansatz, da eine Begrenzung auf eine bestimmte Domäne zugrunde liegt. Das heißt, dass die Gruppe von Texten, die zusammengefasst werden können, stark begrenzt ist. Beispiele dafür sind Wetterberichte, Sportergebnisse, Stellenausschreibungen oder Nachrichten über Personalwechsel. Für diese Texte ist die Faktenextraktion sehr leistungsfähig, allerdings ist sie somit nicht sehr flexibel.
Durch die gezielte Extraktion von vorher festgelegten Informationen ist es möglich einen neuen kohärenten Text als Ergebnis der Zusammenfassung zu erhalten. Als Vorgaben, welche Informationen relevant und wichtig sind, kommen domänenspezifische Lexikoneinträge oder Regeln in Form von Templates oder Mustern zum Einsatz.
Da diese Art der Textzusammenfassung mit vielen komplexen Problemen verbunden ist, wird im Implementationsteil dieser Arbeit nur auf den Textextraktionsansatz eingegangen.

2.2.2. Textextraktion

Der historisch ältere und in der Praxis häufig angewendete Ansatz der Textextraktion, auch Keyphrase Extraction genannt, ist ein offener Ansatz, bei dem es vorher keine Informationen über die Art des Inhaltes gibt, da keine Domänenbeschränkungen vorhanden sind. Als wichtig erkannte Textteile werden ausgewählt und ohne weitere Interpretation zu einer Zusammenfassung zusammengefügt. Von Nachteil ist allerdings, dass die dabei entstehenden Texte nur begrenzt kohärent sind, da beim inhaltlichen Sinnzusammenhang sowie bei der Verknüpfung der Sätze große Mängel auftreten können.
Für das Erkennen und Auswählen wichtiger Textfragmente gibt es eine Reihe unterschiedlicher Methoden, welche oft auch kombiniert eingesetzt werden.

2.2.2.1. Extraktionsmethoden

Den unterschiedlichen Extraktionsmethoden ist gemein, dass sie in 3 Schritten ablaufen:

1. Relevante Textteile identifizieren
2. Minimale Textteile mit relevanten Textsegmenten aus dem Originaltext extrahieren
3. Diese Textteile zu einer Zusammenfassung zusammenstellen

Der Unterschied liegt im Verfahren zur Identifikation relevanter Textteile.

2.2.2.1.1. Statistische Extraktionsmethoden

Bei statistischen Extraktionsmethoden wird für jeden Satz ein Wert für dessen Relevanz berechnet. Dies geschieht über unterschiedliche Methoden (Schlüsselwort-Methode, Positionsmethode, …). Die Ergebnisse dieser unterschiedlichen Methoden werden dann „von Hand gewichtet und in eine Linearformel integriert“ [Endres-Niggemeyer 2002], um so die Wichtigkeit eines Satzes zu bestimmen. Dann werden die Sätze mit den größten Werten in die Zusammenfassung übernommen.

2.2.2.1.1.1. Schlüsselwort-Methode

In der Annahme, dass ein Autor in inhaltlich wichtigen Sätzen bestimmte Schlüsselwörter verwendet, welche seine Ideen und Meinungen ausdrücken, sucht diese Methode nach diesen Schlüsselwörtern und extrahiert Textsegmente, die diese enthalten. Von dieser Methode gibt es unterschiedliche Varianten:

- für jeden Satz werden die in ihm vorkommenden Schlüsselwörter gezählt und gespeichert; die Zusammenfassung entsteht aus der Auswahl der Sätze mit den größten Werten
- die in der Überschrift enthaltenen Worte werden als Schlüsselworte betrachten und es werden die Sätze ausgewählt, die diese Schlüsselworte enthalten
- bei nutzerorientierten oder auf Anfragen basierenden Zusammenfassungen werden die Nutzerinteressen oder die Anfrage-Worte als Schlüsselworte benutzt
- es kommen nur Substantive oder Verben als Schlüsselworte in Frage, da diese bedeutsamer sind

Natürlich können auch Kombinationen dieser Varianten eingesetzt werden.

2.2.2.1.1.2. Positionsmethode

Bei dieser Methode wird davon ausgegangen, dass die Position bestimmter Sätze etwas über die Wichtigkeit dieser Sätze aussagt. So gelten Sätze in der Nähe von Überschriften oder in der Nähe des Beginns oder des Endes von Dokumenten, Kapiteln oder Absätzen als wichtig.

2.2.2.1.1.3. Methode basierend auf Hinweisphrasen

Diese Methode basiert auf der Annahme, dass der Autor relevante Textsegmente durch darauf hinweisende Phrasen einleitet. Solche anzeigenden Ausdrücke können sein: „wichtig ist, dass“, „abschließend“, „das Thema dieser Arbeit“, „Ergebnisse“ oder „zusammenfassend kann gesagt werden, dass“. Sätze oder Satzteile, die diese Wendungen enthalten, werden in die Zusammenfassung übernommen. Dabei muss beachtet werden, dass die hinweisenden Formulierungen abhängig vom Genre des Textes sein können. In Wetternachrichten können zum Beispiel Worte und Wortgruppen wie „Höchsttemperatur“, „Aussichten für morgen“ die interessanten Sätze oder Satzsegmente einleiten. Im Gegensatz dazu sind bei Nachrichten über Sportereignisse Hinweisworte und –wortgruppen wie „der Gewinner ist“, „Punktestand“ oder „Bestzeit“ von Interesse.
So wie bestimmte Wendungen als Auswahlkriterium dienen, können auch andere Wortgruppen wie „nicht wichtig, dass“, „nicht ausschlaggebend“, „unmöglich“ oder „nebensächlich“ als Ausschlusskriterium für die Sätze gelten, die diese Worte enthalten.

2.2.2.1.1.4. Methode der Worthäufigkeiten

Die Methode der Worthäufigkeiten, auch Key-Method genannt, ist ein früher, sehr schwacher Ansatz. Bei dieser Methode wird von der Worthäufigkeit auf die Wortsignifikanz geschlossen. Das heißt, dass man annimmt, dass für die Zusammenfassung relevante Sätze Worte enthalten, die häufig im gesamten Text auftreten. Damit diese Methode einigermaßen sinnvoll verwendet werden kann, müssen nicht nur sehr selten vorkommende Worte ignoriert werden, sondern auch die am häufigsten vorkommenden Worte. Dieses sind meist grammatikalische Füllwörter wie Artikel (der, die, das, ein, …), Konjunktionen (oder, und, wie, sowohl, …), Präpositionen (zu, in, vor, hinter, …), Fragewörter (wer, wie, wann, was,…) oder Modalverben (wollen, haben, machen, …). Deshalb gelten nur die Worte als relevant, welche zwischen einem oberen und unteren Grenzwert liegen (vgl. Abbildung 1).

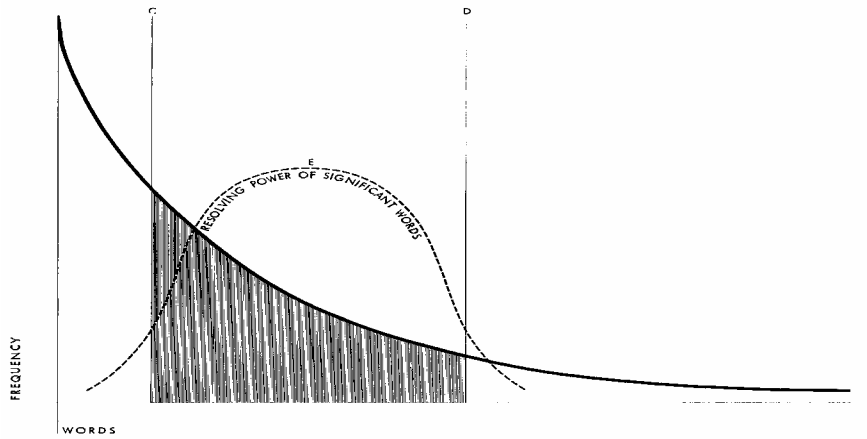


Abbildung 1: Wortfrequenzdiagramm [Luhn 1958]

2.2.2.1.1.5. Satzlängen-Methode

Ein anderer sehr einfacher Ansatz ist die Satzlängen-Methode. Hier wird davon ausgegangen, dass Zusammenfassungen, die längere Sätze enthalten, besser sind als Zusammenfassungen mit kurzen Sätzen.

2.2.2.1.2. Extraktionsmethoden basierend auf dem Einsatzbereich

Bei diesen Methoden wird Wissen über die Domäne des Textes ausgenutzt. So gibt es Verfahren, die berücksichtigen, dass die Texte einer Domäne, die zusammengefasst werden sollen, stets eine ähnliche Struktur haben. Bei wissenschaftlichen Texten kann davon ausgegangen werden, dass am Anfang ein Abstrakt, der einen kurzen Überblick über den Text gibt, und am Ende meist eine Zusammenfassung und Schlussfolgerung steht. Mit diesem Wissen kann das Extraktionssystem gezielt Sätze aus diesen Textteilen extrahieren und sicher sein, die wichtigsten Sätze erwischt zu haben.
Andere domänenbasierte Methoden nutzen Skripts, Schemata oder Templates und funktionieren ähnlich den Verfahren zur Faktenextraktion.
Des Weiteren kommen auch statistische Methoden, die an die Domäne angepasst wurden, zum Einsatz.

2.2.2.1.3. Weitere Extraktionsmethoden

2.2.2.1.3.1. Relationale Methode

Diese Methode extrahiert unter der Annahme, dass Sätze, die sehr stark zu anderen Sätzen in Bezug stehen, sehr wichtig für den gesamten Text sein müssen. Die Schlussfolgerung daraus ist, dass solche Sätze in die Zusammenfassung übernommen werden sollten. Denn das Weglassen von Sätzen, die sich stark auf andere beziehen oder auf die sich andere Sätze beziehen, hätte zur Folge, dass kein zusammenhängender Text entsteht. Das Aufspüren von in Beziehung stehenden Sätzen benötigt einen Thesaurus bzw. ein Wortnetz und ist nicht trivial.

2.2.2.1.3.2. Text-Mining-basierte Methode

Die Idee dieser vom Information Retrieval inspirierten Methode ist, dass Sätze, die sehr repräsentative Worte enthalten, wichtig für den Text sind und dass je seltener die Worte eines Satzes in den anderen Sätzen vorkommen, umso repräsentativer sind diese für den Satz. Diese Methode nennt sich Term Frequency – Inverse Sentence Frequency und wurde von J. Larocca Neto, A. D. Santos, A. A. Kaestner und A. A. Freitas vorgeschlagen [Larocca Neto, Santos, Kaestner & Freitas 2000].

2.3. Entwicklung der Automatischen Textzusammenfassung

Die Geschichte der Automatischen Textzusammenfassung begann in den 1950er Jahren mit den ersten Extraktionsalgorithmen. Später folgten Ansätze, die sprachwissenschaftliches Wissen ausnutzten. In den 1980er Jahren bediente man sich für die Automatische Textzusammenfassung des Wissens und der Erkenntnisse der Künstlichen Intelligenz. In den 1990er Jahren erlebte die Forschung auf dem Gebiet der Automatischen Textzusammenfassung eine Renaissance. Durch die riesigen Informationsmengen, welche plötzlich im Internet zu finden waren, stieg der Bedarf an Systemen, die helfen, wirklich relevante Informationen in möglichst kurzer Zeit zu finden.

Es sollen nun die wichtigsten Algorithmen und Ideen der Automatischen Textzusammenfassung von den 50er Jahren bis heute kurz vorgestellt werden.

2.3.1. Die Anfänge der Forschung

2.3.1.1. Der Luhn-Algorithmus (1958)

Luhns Arbeit „Automatic Creation of Literature Abstracts“ von 1958 beschreibt die erste Implementierung eines Satzextraktionsalgorithmus. Als sinnvolles Maß für die Relevanz von Worten sah Luhn die Häufigkeit, mit der ein Wort im Text auftaucht. Die Berechtigung für diese Annahme sieht er darin, dass ein Autor bestimmte Worte, die mit dem Thema verbunden sind, bei seiner Argumentation und der Beschreibung verschiedener Aspekte wiederholt. Außerdem war er der Meinung, dass die Position von relevanten Worten innerhalb eines Satzes etwas über die Wichtigkeit dieses Satzes aussagt. Aus einer Kombination dieser beiden Werte wollte Luhn die Relevanz der Sätze bestimmen.
Es wird also als Erstes eine Art „Inventarliste“ mit allen vorkommenden Worten und deren Häufigkeit erstellt. Da Luhn der Meinung ist, dass nur Worte mit mittlerer Häufigkeit etwas über die Signifikanz eines Satzes aussagen und Worte mit sehr hoher Häufigkeit eher nichts sagend, weil zu allgemein sind, will er diese allgemeinen Worte mit sehr hohem Vorkommen im Text ausschließen. Er sieht zwei Möglichkeiten dies zu tun:

- Vergleich dieser hochfrequenten Worte mit einer Liste mit allgemeinen Worten und Ausschluss der Worte, die als allgemein gelten, aus der Berechnung der Relevanz
- Festlegen eines oberen und eines unteren Grenzwertes bezüglich der Häufigkeit, um zu allgemeine und zu selten vorkommende Worte auszuschließen

Luhn untersuchte beide Varianten. Um die optimalen Grenzwerte zu finden, musste man sich auf die Erfahrung aus vielen Beispielartikeln verlassen. (vgl. Abbildung 1)
Der Signifikanzwert eines Satzes berechnet sich aber nicht einfach aus den enthaltenen relevanten Worten. Da Luhn die Position und die Beziehung relevanter Worte auch berücksichtigen wollte, sollten nur Satzteile, welche relevante Wortgruppen enthielten, einbezogen werden. Es wurde festgelegt, dass ein relevantes Wort nur dann zu so einer Wortgruppe (genannt Cluster) gehört, wenn zwischen ihm und dem nächsten relevanten Wort nicht mehr als vier oder fünf unwichtige Worte stehen. Der Signifikanzfaktor berechnet sich daher wie folgt:

$$\text{Signifikanzfaktor} = \frac{(\text{Anz. signifikanter Worte der Wortgruppe})^2}{\text{Anz. Worte der Wortgruppe}}$$

Nachdem die Sätze entsprechend ihrer Relevanz geordnet wurden, sollte der Satz bzw. die Sätze mit den höchsten Relevanzwerten für die Zusammenfassung ausgewählt werden.

2.3.1.1.1. Luhns Bilanz

Laut Luhn, zeigten die Ergebnisse, das heißt die automatisch generierten Extrakte, dass es mit seinem Algorithmus möglich war Zusammenfassungen automatisch zu erstellen, die das Hauptthema des Originals fast genau so gut wiedergaben wie herkömmliche Zusammenfassungen. Ein Vorteil der so erstellten Zusammenfassungen war laut Luhn ihre Zuverlässigkeit, Konsistenz und Beständigkeit, was daher rührte, dass die unterschiedlichen Fähigkeiten und Orientierungen von Menschen keinen Einfluss auf die Zusammenfassung hatten. Nach Luhns Meinung würden die Nutzer von Zusammenfassungssystemen nach und nach Lernen wie die erstellten Zusammenfassungen interpretiert werden müssen. So würden die Nutzer erkennen, dass einige Worte sich auf Bemerkungen aus vorangegangenen, nicht extrahierten Sätzen beziehen.
Er sah aber auch einige Nachteile, die die automatisch generierten Zusammenfassungen mit sich brachten. Er nannte zum Beispiel den Verlust der Gewandtheit der Zusammenfassungen. Auch sah er Probleme, wenn der Stil eines Autors stark von der Allgemeinheit abweicht, da so eventuell geringer-wertige Sätze ausgewählt werden könnten.
Trotz der Nachteile war Luhn der Meinung, dass mit der automatischen Erstellung von Zusammenfassungen beachtliche und lohnende Einsparungen des menschlichen Aufwands erreicht werden könnten (vgl. [Luhn 1958])
Allerdings erkannte er auch Möglichkeiten seinen Algorithmus zu verbessern. Zum einen könnte sein Ansatz dahingehend geändert werden, dass Zusammenfassungen von Texten zu bestimmten Themen oder Untersuchungsbereichen entstehen. Zum anderen sah er Bedarf, Zusammenfassungen mit variabler Länge generieren zu lassen. So könnten zum Beispiel Zusammenfassungen entstehen, die auf die Bedürfnisse des einzelnen Anwenders zugeschnitten sind. Sollten dann die Signifikanzwerte der einzelnen Sätze nicht über einen bestimmten Grenzwert hinaus kommen, kann der Artikel als „zu allgemein“ für die Nutzerinteressen abgewiesen werden.

2.3.1.1.2. Nachteile des Luhn-Algorithmus

Ronald E. Wyllys betrachtete als großen Nachteil des Algorithmus von Luhn, dass dieser nur die absolute Häufigkeit der einzelnen Worte als Relevanzkriterium verwendete. Er war der Meinung, dass allgemeine Worte wie „importance“, „connection“ oder „theory“ sehr häufig in

Texten auftauchen ohne eine vergleichbar große Relevanz für das Thema des Textes zu haben (vgl. [Wyllys 1967, S. 162]).
Als Lösungsmöglichkeit für dieses Problem schlug Wyllys vor, solche Worte in die Liste der allgemeinen Worte aufzunehmen und diese somit von der Häufigkeitsberechnung auszuschließen.
Als weiteren Nachteil der bloßen Berücksichtigung der absoluten Häufigkeit sah Wyllys die Tatsache, dass hochspezialisierte Terme, welche wichtig für den Inhalt des Dokumentes sind, nicht oft genug im Text erwähnt werden, um nach der Häufigkeitszählung als relevant zu gelten.

2.3.1.2. Berücksichtigung struktureller und linguistischer Textcharakteristika – Algorithmus von Edmundson (1969)

Edmundson orientierte sich bei seinem Algorithmus zur Automatischen Textzusammenfassung mehr am Vorgehen der Menschen beim Schreiben einer Zusammenfassung. Denn seiner Meinung nach konnten mit Luhns rein statistischem Ansatz zwar Zusammenfassungen mit ausreichender Qualität für eine Weiterführung der Forschung erzielt werden, aber diese Zusammenfassungen waren von mangelhafter Qualität.
Er setzte mit seinem Extraktionsalgorithmus den Trend bei der Satzextraktion. Er untersuchte Kombinationen von verschiedenen linguistischen und strukturellen Eigenschaften.
Um herauszufinden nach welchen Textmerkmalen sich Menschen richten, wenn sie eine Zusammenfassung erstellen, und um diese für seinen Algorithmus nutzen zu können, ging Edmundson in folgenden Schritten vor:

1. Auswahl eines Dokumentenkorpus
2. Analyse von Merkmalen traditioneller Zusammenfassungen von Dokumenten hinsichtlich des Inhalts und der Wiedererkennbarkeit durch den Computer
3. Spezifizierung der gewünschten Form und des gewünschten Inhalts an Hand der manuellen Zusammenfassungen
4. Herstellen von Zusammenfassungen für einige Testdokumente, die sich nach der Spezifizierung des Inhalts und der Wiedererkennbarkeit durch den Computer richten
5. Erstellen eines Systems, das den Merkmalen, die der Computer wieder erkennt, numerische Gewichte zuweist

6. Einen Computer so programmieren, dass er automatische Zusammenfassungen erstellt
7. Periodisches Verbessern des Verfahren durch den Vergleich der automatischen Zusammenfassung mit von Hand erstellten
8. Evaluierung des Extraktionsalgorithmus mit Hilfe von neuen, vorher noch nicht verwendeten Dokumenten

Die schwierigste Aufgabe war es, Textcharakteristika zu finden, die einerseits vom Computer wieder erkannt werden (Vorhandensein bestimmter Worte, Position eines Satzes, die Länge des Satzes) und die andererseits einen Satz als relevant oder nicht relevant deklarieren. Dafür wurden Textmerkmale als „positiv relevant“ bezeichnet, wenn sie in Sätzen auftraten, die manuell für die Zusammenfassung ausgewählt worden wären. „Negativ relevant“ bezeichnete man Merkmale, die in Sätzen gefunden wurden, die nicht manuell für eine Zusammenfassung ausgewählt worden wären. Merkmale, die in beiden Satztypen auftauchten, wurden als „irrelevant“ bezeichnet.
Edmundson fand so vier verschiedene Textcharakteristika, die für die Selektionsentscheidung ausgewertet wurden, und erstellte daraus vier Methoden, mit denen die vom Computer wieder erkennbaren Merkmale gewichtet werden konnten. Die Summe der einzelnen Merkmalsgewichte stellte das Gewicht des gesamten Satzes dar.

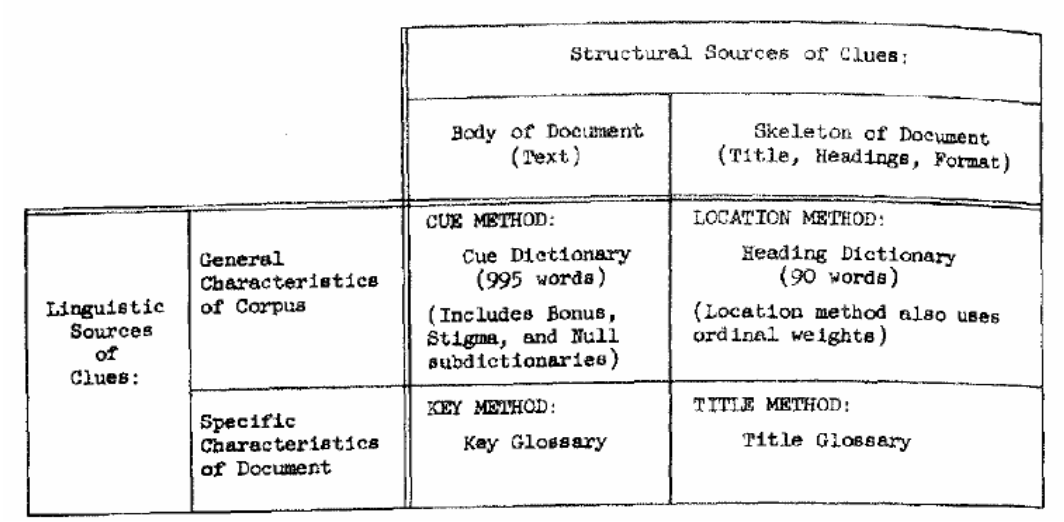
2.3.1.2.1. Die vier Methoden zur Satzgewichtung

- *Cue-Methode*: Die Cue-Methode (Cue = Stichwort, Hinweis, Wink) geht von der Annahme aus, dass die Relevanz eines Satzes von dem Vorkommen von pragmatischen Worten wie „signifikant“, „unmöglich“ oder „kaum“ abhängt. Dafür wurde ein Wörterbuch mit den drei Unterbereichen „Bonus-Wort“, „Stigma-Wort“ und „Null-Wort“ aus dem Korpus erstellt und verwendet. Das Cue-Gewicht eines Satzes stellte die Summe der Cue-Gewichte der wesentlichen Worte des Satzes dar.
- *Key-Methode*: Das Prinzip der Key-Methode entspricht im Großen und Ganzen dem Ansatz von Luhn, allerdings nicht dessen Algorithmus. Dabei wird für jedes Dokument ein Key-Glossar erstellt, welches die Themenworte des Dokuments enthält. Das Glossar wird aus den Worten gebildet, die nicht im Cue-Wörterbuch enthalten sind, wobei diese absteigend ihrer Häufigkeit geordnet sind. Alle diese Nicht-Cue-Worte, deren Häufigkeit über einem Grenzwert liegt, erhalten positive Gewichte entsprechend

ihrer Häufigkeit. Auch hier ist das Key-Gewicht eines Satzes die Summe der Key-Gewichte der wesentlichen Worte des Satzes.

- *Title-Methode*: Die Anhaltspunkte für die Satzrelevanz sind bei der Title-Methode die Merkmale des Dokumentenskeletts, das heißt der Titel, Zwischenüberschriften und Formatierungen. Die Grundlage dieser Methode ist die Annahme, dass der Autor mit dem Titel das Thema des Dokuments umschreibt und dass er bei der Einteilung des Dokuments in Abschnitte diese Abschnitte mit den Zwischenüberschriften zusammenfasst. Die Worte des Titels und der Überschriften gelten somit als „positiv relevant“ und werden deshalb in ein Titelglossar für das Dokument übernommen und erhalten positive Gewichte, wobei Titelworte mehr Gewicht erhalten als Worte aus Zwischenüberschriften. Aus den Title-Gewichten der einzelnen Worte eines Satzes bildet man durch Addition dieser das Title-Gewicht des Gesamtsatzes.
- *Location-Methode*: Die Location-Methode nimmt die Überschriften und die Formatierung eines Dokumentes als Kriterium für die Relevanz der Sätze. Dabei wird von folgenden zwei Hypothesen ausgegangen:
    1. Sätze, die unter bestimmten Überschriften (bspw. „Einleitung“, „Zusammenfassung“) stehen, sind positiv relevant
    2. Für das Thema relevante Sätze erscheinen relativ am Anfang oder relativ am Ende des Abschnitts oder Dokuments

  Die erste Annahme wurde durch Vergleich der Übereinstimmungen mit den manuell ausgewählten Sätzen überprüft und es wurden aus der Statistik abgeleitete Gewichte für Sätze nach bestimmten Überschriften vergeben. Außerdem erhielten Sätze anhand ihrer Position im Dokument Gewicht. So bekamen Sätze des ersten und des letzten Abschnitts sowie der erste und letzte Satz eines Dokuments positive Gewichte. Das Location-Gewicht eines Satzes errechnet sich daher aus dem Überschriften-Gewicht und dem Positionsgewicht.



		Structural Sources of Clues:	
		Body of Document (Text)	Skeleton of Document (Title, Headings, Format)
Linguistic Sources of Clues:	General Characteristics of Corpus	CUE METHOD: Cue Dictionary (995 words) (Includes Bonus, Stigma, and Null subdictionaries)	LOCATION METHOD: Heading Dictionary (90 words) (Location method also uses ordinal weights)
	Specific Characteristics of Document	KEY METHOD: Key Glossary	TITLE METHOD: Title Glossary

Abbildung 2: Verschiedene Quellen der Anhaltspunkte für die Relevanzentscheidungen und ihr Zusammenspiel in den verschiedenen Methoden[Edmundson 1969]

2.3.1.2.2. Ergebnisse der Experimente von Edmundson

Anhand einer Vielzahl von Experimenten suchte Edmundson nach der besten Methode oder Methoden-Kombination, indem er die prozentuale Übereinstimmung zwischen den manuell erstellten Zusammenfassungen und den automatisch generierten Zusammenfassungen ermittelte.

Dabei errechnete sich das Satzgewicht bei der Nutzung mehrerer Methoden folgendermaßen:

$$Gewicht_{Satz} = a_1 * Gewicht_{Cue} + a_2 * Gewicht_{Key} + a_3 * Gewicht_{Title} + a_4 * Gewicht_{Location}$$

wobei $a_1 ... a_4$ positive Integer-Werte sind.

Außerdem wurde die Übereinstimmung zwischen den manuell erstellten Beispiel-zusammenfassungen und Zufallszusammenfassungen zum Vergleich berechnet (siehe Abbildung 3).

Edmundson konnte so feststellen, dass mit der Cue-Title-Location-Kombination die besten Ergebnisse erzielt werden konnten und dass die bloße Ausnutzung der Key-Methode die schlechtesten Ergebnisse für die automatisch erstellten Zusammenfassungen lieferte und sie daher nicht in das Zusammenfassungssystem übernommen werden sollte.

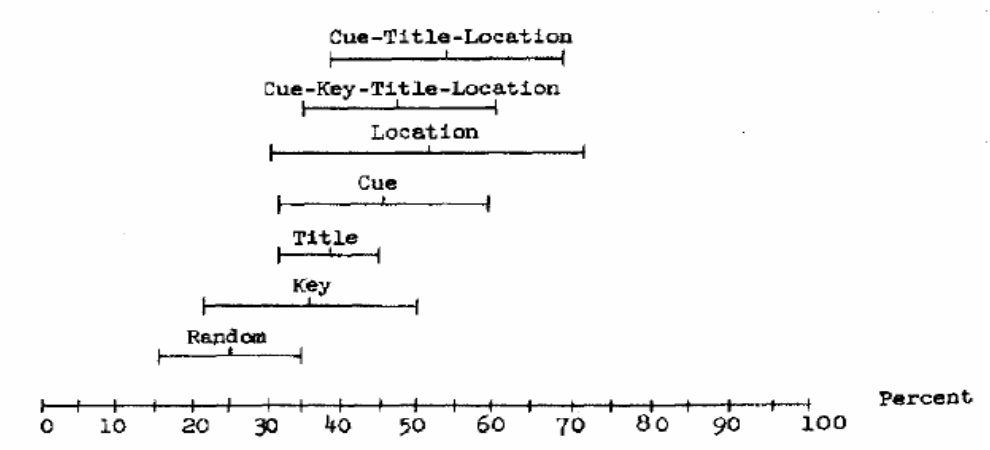


Abbildung 3: Durchschnittliche Übereinstimmungsbereiche (in %) mit der Beispielzusammenfassung der unterschiedlichen Methoden und Methoden-Kombinationen

2.3.1.2.3. Edmundsons Bilanz

Edmundson sah seinen Ansatz und die damit erzielten Ergebnisse als erfolgreich und mit weiterer Forschung ausbaubar. So könnten weitere Experimente die verschiedenen Gewichte dahingehend verbessern, dass noch bessere Zusammenfassungen erstellt werden können.
Außerdem sah er Verbesserungsbedarf bei der Verarbeitung längerer Dokumente und bei der Verringerung des Zeitbedarfs und der Kosten. Des Weiteren war Edmundson der Ansicht, dass der Prozess der Texteingabe in den Rechner verbessert werden sollte, da es zum Beispiel nicht möglich war chemische und mathematische Symbole automatisch mit einzulesen. Ferner sah er Elemente mit möglicher Relevanz, die bisher völlig unbeachtete geblieben waren und die in Zukunft in den Extraktionsentscheidungen berücksichtigt werden sollten, wie beispielsweise Zeichnungen, Bilder, Schemata, Tabellen, Fußnoten oder Referenzen.

2.3.2. Die Renaissance in der Forschung seit den 1990ern

2.3.2.1. Ein trainierbarer Textzusammenfasser

Julian Kupiec, Jan Pedersen und Francine Chen vom Xerox Palo Alto Research Center entwickelten 1995 ein trainierbares Dokumentenzusammenfassungssystem. Die Motivation für die Entwicklung dieses extraktiv-arbeitenden Textzusammenfassers sahen die Forscher in der Schaffung eines einfachen, informativen Zwischenstücks zwischen dem Titel eines Dokuments und dessen Vollversion, welches Relevanzentscheidungen enorm beschleunigen sollte. Ihr Ziel war es daher eine knappe aber präzise Dokumentenbeschreibung zu generieren, welche aufschlussreicher als der Titel sein sollte und zugleich in einem Blick erfasst werden konnte.

2.3.2.1.1. Extraktionsmerkmale

Für eine optimale Satzbewertung wurden von Kupiec, Pedersen und Chen unterschiedliche, potentielle Bewertungskriterien getestet, kombiniert und mit Hilfe eines Korpus von Volltexten und den dazugehörigen Zusammenfassungen gewichtet.

- *Sentence Length Cut-off Feature*: Es werden nur Sätze ab einer gewissen Länge (z.B. mind. 5 Worte) in Zusammenfassungen übernommen. Ist ein Satz länger als dieser Grenzwert, ist dieses Feature wahr, sonst ist es falsch.
- *Fixed-Phrase Feature*: Es werden Sätze in die Zusammenfassung übernommen, wenn in ihnen mindestens eine Wortgruppe aus einer Liste von Wortgruppen (wie „this letter“, „in conclusion“ usw.) vorkommen oder wenn sie nach Überschriften stehen, die Worte wie „conclusion“, „results“ oder „summary“ enthalten.
- *Pharagraph Feature*: Dieses Feature wird auf die ersten 10 und die letzten 5 Paragraphen angewendet. Sätze in einem Paragraphen werden als „paragraph-initial“, „paragraph-final“ und „paragraph-medial“ bewertet.
- *Thematic Word Feature*: Es werden einige wenige so genannte „thematic words“ ausgewählt. Das sind die am häufigsten vorkommenden Inhaltsworte. Außerdem werden alle Sätze mit einer Häufigkeitsfunktion bewertet. Ist ein Satz in der Menge der am höchsten bewerteten Sätze enthalten ist dieses Feature wahr.
- *Uppercase Word Feature*: Ausgehend von der Annahme, dass Eigennamen meist wichtig sind, wird dieses Feature ähnlich dem vorhergehenden berechnet, allerdings mit der Beschränkung, dass ein „uppercase thematic word“ nicht am Anfang des Satzes steht, mit einem Großbuchstaben beginnt, mehrmals im Text vorkommt und keine Maßeinheit ist. Des Weiteren werden die Sätze doppelt so wichtig bewertet, in denen ein „uppercase thematic word“ das erste Mal auftritt.

Mit Hilfe dieser Feature, dem Textkorpus und der Bayesschen Regel berechneten Kupiec, Pedersen und Chen die Wahrscheinlichkeit, dass der Satz s unter Beachtung der k Features F_1 bis F_k in die Zusammenfassung S aufgenommen wird, wie folgt:

$$P(s \in \mathcal{S}|F_1, F_2, \ldots F_k) = \frac{P(F_1, F_2, \ldots F_k|s \in \mathcal{S})P(s \in \mathcal{S})}{P(F_1, F_2, \ldots F_k)}$$

$$P(s \in \mathcal{S}|F_1, F_2, \ldots F_k) = \frac{\prod_{j=1}^{k} P(F_j|s \in \mathcal{S})\ P(s \in \mathcal{S})}{\prod_{j=1}^{k} P(F_j)}$$

Abbildung 4: Gleichungen zur Wahrscheinlichkeitsberechnung [Kupiec 1995, S. 69]

Dabei war $P(s \in S)$ eine Konstante und $P(F_j | s \in S)$ sowie $P(F_j)$ konnten aus dem Trainingskorpus durch Zählen der Vorkommen berechnet werden.
Der Trainingskorpus bestand aus 188 Dokument-Zusammenfassungs-Paaren aus dem wissenschaftlichen und technischen Bereich.

2.3.2.1.2. Die Ergebnisse von Kupiec, Pedersen und Chen

Um die Übereinstimmung zwischen der automatisch erstellten Zusammenfassung und der manuell erstellten abstraktiven Zusammenfassung in der Trainings- und Evaluationsphase feststellen zu können, wurde nach Übereinstimmungen zwischen den Sätzen der manuellen Zusammenfassungen und dem Original gesucht und diese in verschiedene Matching-Klassen eingeteilt.

- *Direct Sentence Match*: In dieser Klasse findet man Sätze der manuellen Zusammenfassung, die wortwörtlich oder mit geringfügigen Änderungen aus dem Original extrahiert wurden.
- *Direct Join*: Als Direct Join klassifiziert man die Zusammenfassungssätze, die aus zwei oder mehr Sätzen des Originals gebildet wurden.
- *Unmatchable*: In diese Klasse gehören Zusammenfassungssätze, die der Zusammenfasser nach einem allgemeinen Lesen des Originals und ohne das Benutzen eines bestimmten Originalsatzes erstellt hat.
- *Incomplete Sentence Match*: Es gehören zwei Arten von Zusammenfassungssätzen in diese Gruppe: wenn ein Zusammenfassungssatz einige Überlappungen mit einem Originalsatz hat, aber nicht der gesamte Satzinhalt enthalten ist, oder wenn ein Zusammenfassungssatz neben dem kompletten Inhalt eines Originalsatzes noch zusätzliche Informationen enthält, diese aber nicht von einem Join abgedeckt werden.
- *Incomplete Join:* In diese Kategorie gehören Joins für die dasselbe gilt wie für Sätze der Klasse Incomplete Sentence Match.

Sätze der letzten 3 Klassen wurden sowohl für die Trainingsphase als auch für die Testphase aus den Zusammenfassungen entfernt.
Die Evaluierung erfolgte auf zwei Arten. Als erstes wurde der Anteil der Sätze der manuellen Zusammenfassungen, die durch das Zusammenfassungssystem reproduziert wurden, gemessen. Dabei galt ein automatisch generierter Zusammenfassungssatz als korrekt, wenn er entweder einen Direct Sentence Match in der manuellen Zusammenfassung hatte oder wenn er Teil eines Direct Joins der manuellen Zusammenfassung war, von dem auch der Rest in der automatischen Zusammenfassung reproduziert wurde. Auf diese Art und unter der Annahme

der Existenz nur einer korrekten Zusammenfassung konnte ermittelt werden, dass 35 % der Informationen der manuellen Zusammenfassungen durch die automatisch generierten Zusammenfassungen wiedergegeben werden. Dabei wurden Zusammenfassungen automatisch generiert, die die gleiche Länge (3 Sätze) wie die manuell erstellten Zusammenfassungen hatten.

Als zweiter Teil der Evaluierung wurden die einzelnen Features getestet. Es zeigte sich, dass die Kombination von Paragraph-Feature, Fixed-Phrase Feature und Sentence Length Cut-off Feature die besten Ergebnisse (44 %) lieferte. Das Hinzufügen der Features, welche auf Worthäufigkeiten beruhten, brachte eine leichte Performanzabnahme mit sich (nur noch 42 %). Trotzdem wurden diese Features im Endsystem beibehalten, da sie für eine gewisse Robustheit sorgen, vor allem dann, wenn zum Beispiel keine Indikatorwortgruppen im Text enthalten sind, und da sie bei größeren Zusammenfassungen verteilte Informationen finden.

Bei der Verwendung nur eines Features waren die Ergebnisse des Paragraph-Features die besten (33 %) und die Ergebnisse des Thematic-Word-Features bzw. des Uppercase-Word-Features (je 20 %) die schlechtesten.

Bei einem weiteren Test bei dem unterschiedliche Zusammenfassungsgrößen verwendet wurden, ermittelten Kupiec, Pedersen und Chen, dass für Zusammenfassungen, die ein Viertel so groß sind wie das Original, 84 % der korrekten Sätze durch das Zusammenfassungssystem ausgewählt wurden. Für etwas kürzere Zusammenfassungen wurden immerhin noch 74 % der korrekten Sätze ausgewählt.

2.3.2.2. Das SUMMARIST – Textzusammenfassungssystem

Eduard Hovy und Chin-Yew Lin entwickelten von 1995 - 1997 das robuste Textzusammenfassungssystem SUMMARIST an der Universität von Southern California. Ausgegangen sind sie dabei von folgender Gleichung:

> „summarization = topic identification + interpretation + generation“
>
> [Hovy & Lin 1999, S. 81]

Die einzelnen Komponenten der Gleichung wurden in separaten Modulen implementiert, welche alle verschiedene Wissensquellen nutzen.

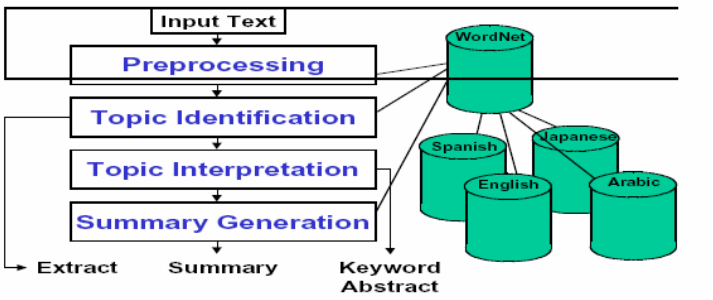


Abbildung 5: Architektur von SUMMARIST [Hovy & Lin 1999, S. 84]

Bevor der Text die einzelnen Stationen durchlaufen konnte, musste er in ein standardisiertes, internes Format konvertiert werden. Auch für diesen Schritt wurden verschiedene Module implementiert:

- *Tokenizer*: Zerlegt den englischen Text in die einzelnen Token (Worte).
- *Part-Of-Speech Tagger*: Wandelt zerlegten Text in part-of-speech-markierten Text um.
- *Converter*: Konvertiert den part-of-speech-markierten Text in das interne SUMMARIST-Format.
- *Morpher*: Ermittelt die Stammformen der einzelnen Token.
- *Phraser*: Findet feststehende Wortgruppen, welche in WordNet hinterlegt sind.
- *Token Frequency Counter*: Bestimmt die Häufigkeiten der einzelnen Token im Originaltext.
- *tf.idf Weight Calculator*: Bestimmt das tf.idf-Gewicht für jedes Token und ordnet die Token entsprechend ihrer Gewichte.
- *Query Relevance Calculator*: Ermittelt satzweise die Anzahl der Worte der Nutzeranfrage, die auch in dem Satz enthalten sind.

2.3.2.2.1. Die SUMMARIST – Themenidentifikation

Die Ermittlung der relevantesten Sätze, hier als Topic Identification bezeichnet, wurde in SUMMARIST mit Hilfe unterschiedlicher Module, welche verschiedene Methoden benutzten, ausgeführt. Die Ergebnisse der einzelnen Module wurden am Ende kombiniert, so dass alle Sätze entsprechend ihres Gewichtes geordnet werden konnten. Die Module nutzten Methoden basierend auf:

- der Position des Satzes
- der Häufigkeit enthaltener Worte
- enthaltenen Hinweis-Phrasen.

2.3.2.2.1.1. Themenidentifikation mit dem Positionsmodul

Dieses Modul nutzt den Fakt, dass bei einigen Textgenres an bestimmten Positionen mit themenrelevanten Sätzen zu rechnen ist. Dafür definierten Lin und Hovy die Optimal Position Policy (OPP), welche in Listenform die Positionen von relevanten Sätzen enthalten. So wurde zum Beispiel für den Ziff-Davis-Korpus von 130000 Zeitungsartikeln über Computerprodukte folgende OPP aufgestellt:

[T1, P2S1, P3S1, P4S1, P1S1, P2S2, {P3S2, P4S2, P5s1, P1S2}, P6S1, …]

Abbildung 6: OPP für Ziff-Davis-Korpus (T = Titel, P = Paragraph, S = Satz) [Hovy & Lin 1999, S. 85]

2.3.2.2.1.2. Themenidentifikation mit dem Cue-Phrase-Modul

Dieses Modul belohnt Sätze, in denen Hinweisphrasen wie „this paper“, „this document“, „paper gives“, „this paper presents“ oder „we conclude“ vorkommen, mit einem für die Hinweisphrase konstanten Wert. Diese Hinweisphrasen haben Hovy und Lin auf unterschiedliche Art und Weise ermittelt. Zum einen haben sie diese manuell aus einem Korpus von Zusammenfassungen gewonnen. Zum anderen wurden Hinweisphrasen automatisch aus einem Korpus von 87 Artikeln über „Computational Linguistics“ generiert. Dabei verglich man das Verhältnis zwischen der Häufigkeit eines Wortes in der Zusammenfassung geteilt durch die Häufigkeit des Wortes im dazugehörigen Volltext.

2.3.2.2.1.3. Themenidentifikation mit dem Topic-Id-Signature-Modul

Als Topic Signature bezeichnet man ein Themenwort (= Head der Signature) zusammen mit einer Liste von Wort-Gewicht-Paaren, wobei die Worte mit dem Themenwort in einem gewissen Zusammenhang stehen. Die Idee dahinter ist, dass in einem Text, in dem ein bestimmter Begriff auftaucht, auch mit dem Begriff verbundene Worte mit hoher Wahrscheinlichkeit vorkommen. So erhält jedes Vorkommen eines solchen Schlüsselworts das dazugehörige Gewicht. Das Satzgewicht berechnet sich dann aus der Summe der Gewichte der im Satz enthaltenen Schlüsselworte normalisiert durch die Satzlänge.

2.3.2.2.2. Die SUMMARIST-Themeninterpretation

Die Themeninterpretation ist der schwierigste Teil der automatischen Textzusammenfassung. Sie wird aber nur für Zusammenfassungen in Form von Abstrakten benötigt. In diesem Schritt müssen zwei oder mehr extrahierte Gesprächsgegenstände zu einem Inhaltskonzept „verschmolzen“ werden.

Eine einfache Vorgehensweise ist die Teil-Ganzes-Konstruktion. Zum Beispiel: „wheel“, „chain“, „pedal“, „saddle“, „light“, „frame“ und “handlebar” werden zu dem Thema “bicycle” verschmolzen (vgl. [Hovy & Lin 1999, S. 86]). Meist ist dieser Schritt aber sehr viel komplexer und es wird Wissen über die Welt benötigt, welches nicht im Text enthalten ist.
Das SUMMARIST-System von Lin und Hovy interpretiert das Thema unter Nutzung zweier verschiedener Methoden. Allerdings ist die Themeninterpretation noch sehr rudimentär.

2.3.2.2.2.1. Themeninterpretation durch Zählung der Inhaltskonzepte

Die Idee hinter dieser Methode ist, dass Worte, die häufig vorkommen, wichtig sein müssen. Um das Ganze aber besser nutzbar zu machen als die bloße Zählung von Worten, welche Probleme wie die Nichtbeachtung von Synonymen und Pronomen aufweist, sollen nicht Worte sondern Inhaltskonzepte gezählt werden.
Dazu werden alle Inhaltsworte gezählt und diese Häufigkeiten bei dem zu dem Wort gehörenden Konzept in WordNet hinterlegt. Diese Gewichte werden aufwärts propagiert, so dass jeder Knoten als Gewicht die Summe der Gewichte seiner Kinder erhält. Dann wird der Baum von oben nach unten durchlaufen und für jeden Knoten entschieden, ob er eine geeignete Verallgemeinerung seiner Kinder ist. Dies ist der Fall, wenn das Gewicht des Knotens möglichst gleich aus den Gewichten der Kinder abgeleitet ist. Das heißt, dass kein Kindknoten viel mehr zum Gewicht des Elternknotens beigetragen hat als seine Geschwister.

2.3.2.2.2.2. Themeninterpretation mit Topic Signaturen

Nachdem das Themenidentifikationsmodul einige wichtige Themengebiete herausgefunden hat, identifiziert das Topic-Signature-Interpretationsmodul mit Hilfe seiner Sammlung von Signaturen eine oder mehrere Signaturen, welche diese Themengebiete am besten zusammenfassen. Die Heads dieser Signaturen werden dann als das vereinigte Zusammenfassungskonzept bezeichnet.

2.3.2.2.3. Die SUMMARIST-Zusammenfassungsgenerierung

Als letzter Schritt auf dem Weg zur Zusammenfassung steht die Generierung dieser. Da es unterschiedliche Arten von Zusammenfassungen gibt, soll SUMMARIST in naher Zukunft drei verschiedene Generierungsmodule erhalten. Zurzeit ist nur das Modul für extraktive Zusammenfassungen implementiert, welche die Ausdrücke oder Sätze, die durch die Themenidentifikation ausgewählt wurden, enthalten. Da manchmal keine komplette Zusammenfassung benötigt wird, kann SUMMARIST auch die extrahierten Schlüsselworte

oder die interpretierten Zusammenfassungskonzepte als einfache Liste ausgeben. Zukünftig soll SUMMARIST einen Generator enthalten, der aus Einheiten, die Wortgruppen oder Teilsätze sein können, selbst einfache Sätze erstellt. Geplant ist außerdem ein Modul, welches wohlgeformte, fließende Zusammenfassungen produziert. Grundlage sollen Inhaltskonzepte und damit verbundene Themen sein. Genutzt werden soll der Satzplaner von ISI zusammen mit einem Satzgenerator, wie dem Penman, FUF oder NITROGEN.

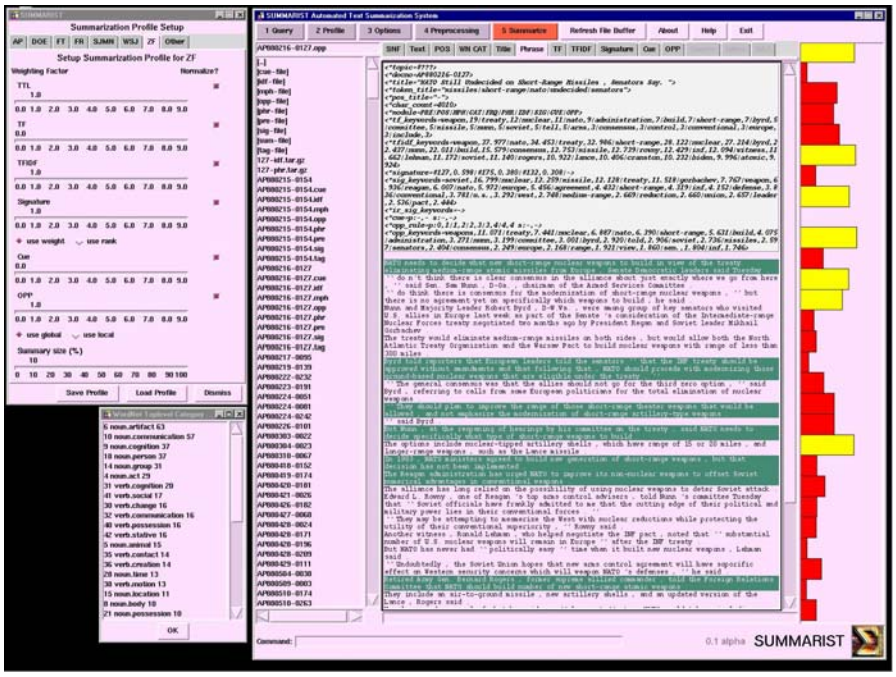


Abbildung 7: Screenshot der Alpha-Version von SUMMARIST

2.3.2.3. Textzusammenfassung unter Nutzung lexikalischer Ketten

Regina Barzilay und Michael Elhadad untersuchten 1997, wie gut sich lexikalische Ketten in der Automatischen Textzusammenfassung nutzen lassen, um die wichtigsten Sätze eines Textes zu ermitteln. Dafür entwickelten sie einen Zusammenfassungsalgorithmus, der in vier Schritten abläuft:

1. Segmentation des Originaltextes
2. Konstruktion der lexikalischen Ketten
3. Identifikation „starker“ Ketten
4. Extraktion signifikanter Sätze

Die Motivation zur Nutzung von lexikalischen Ketten anstatt von Positionsverfahren, Locationverfahren oder Hinweisphrasenentdeckung sahen Barzilay und Elhadad darin, dass

ihr Verfahren nicht vom Textgenre abhängig ist, da es durch die Verarbeitung von lexikalischen Verbindungen und Wortverteilung sich mehr auf den Inhalt des Textes anstatt auf dessen Form bezieht. So waren sie der Meinung, dass ihr System sowohl politische Artikel als auch wissenschaftliche Texte des „Scientic American“ – Magazin zusammenfassen kann. Starke lexikalische Ketten dienen dabei als Anzeiger für das zentrale Thema des Textes.

2.3.2.3.1. Erstellung und Bewertung lexikalischer Ketten

Lexikalische Ketten, welche die Kohäsion im Text anzeigen, entstehen durch die Wiederholung eines Terms, das Auftauchen von Oberbegriffen und Synonymen dieses Terms und durch eine semantische Verwandtschaft von Worten, die sich in häufiger Kookurrenz, das heißt dem häufigen gemeinsamen Auftreten dieser Wörter, äußert.
Die lexikalischen Ketten werden mit Hilfe einer Textsegmentierung, eines syntaktischen Taggers und von WordNet-Relationen erstellt.
Um herauszufinden, welche der erzeugten lexikalischen Ketten für die Erstellung der Zusammenfassung genutzt werden sollten, mussten Barzilay und Elhadad eine Methode finden, die die stärksten Ketten identifiziert. Sie entwickelten dazu eine Umgebung, in der lexikalische Ketten berechnet und visualisiert werden konnten. Mit dieser Umgebung und 30 Texten aus bekannten Magazinen konnten sie experimentell herausfinden, welche Parameter lexikalischer Ketten gute Indikatoren für deren Stärke sind.

- Länge: Die Anzahl von Vorkommen der Mitglieder einer Kette
- Homogenitätsindex: 1 – die Anzahl verschiedener Vorkommen geteilt durch die Länge

Daraus leiteten Barzilay und Elhadad folgende Bewertungsfunktion ab:

Bewertung(Kette) = Länge * Homogenitätsindex

Nachdem die Ketten entsprechend ihrer Bewertung geordnet wurden, ermittelten Barzilay und Elhadad folgendes „Strength Criterion“, welches nur starke Ketten erfüllen:

Bewertung(Kette) > Durchschnitt(aller Bewertungen) + 2 * Standardabweichung(aller Bewertungen)

So ermittelten sie, dass durchschnittlich 5 starke lexikalische Ketten in einem Text vorkamen, wenn durchschnittlich 32 lexikalische Ketten aus einem Text extrahiert wurden.

2.3.2.3.2. Auswahl der wichtigsten Sätze

Nachdem die stärksten Ketten identifiziert waren, mussten Barzilay und Elhadad auf Grundlage dieser signifikante Sätze auswählen. Dazu untersuchten sie drei Varianten:

- 1. Heuristik: Wähle aus jeder starken Kette den Satz, in dem ein Kettenmitglied zum ersten Mal vorkommt.
- 2. Heuristik: Wähle aus jeder starken Kette den Satz, in dem ein repräsentatives Kettenmitglied zum ersten Mal vorkommt. Dabei gilt ein Mitglied als repräsentativ, wenn die Vorkommenshäufigkeit dieses Wortes nicht unter der Durchschnittshäufigkeit dieser Kette liegt.
- 3. Heuristik: Ermittle für jede Kette den Textabschnitt, in dem diese Kette hoch konzentriert ist. Wähle den Satz aus diesem zentralen Abschnitt, in dem die Kette ihr erstes Vorkommen hat.

Diese Heuristiken wurden an den schon erwähnten 30 Texten getestet. Es zeigte sich, dass die dritte Heuristik die schlechtesten Ergebnisse lieferte. Die anderen beiden hatten in den meisten Tests die gleichen Ergebnisse. Waren die Ergebnisse jedoch unterschiedlich, so waren die von der zweiten Heuristik ausgewählten Sätze besser. So entschieden sich Barzilay und Elhadad für diese Methode zur Satzselektion.

2.3.2.3.3. Evaluierung der Zusammenfassungsmethode

Barzilay und Elhadad griffen auf eine intrinsische Evaluation zurück, bei der die Zusammenfassungen, welche ihre Methode erstellt hatte, mit einer idealen, von Menschen erstellten Zusammenfassung verglichen wurden. Außerdem wurde diese Zusammenfassung zum Vergleich Summaries, die der Microsoft Summarizer von Word97 erstellt hatte, gegenübergestellt. Zur Bewertung wurden Precision und Recall berechnet.

	Microsoft		Lexical Chain	
	Prec	Recall	Prec	Recall
10%	33	37	61	67
20%	32	39	47	64

Abbildung 8: Evaluationsergebnisse [Barzilay & Elhadad 1997]

Diese Abbildung zeigt, dass das Zusammenfassungssystem von Barzilay und Elhadad wesentlich bessere Zusammenfassungen produziert als das kommerzielle. Durch diese Resultate konnten Barzilay und Elhadad das große Potential der lexikalischen Ketten als

Wissensbasis für die Satzauswahl beweisen. Allerdings erkannten Barzilay und Elhadad auch, dass ihre Methode einige Nachteile aufweist. Sie waren der Meinung, dass das Selektieren kompletter Sätze ein Hauptproblem ist, da so lange Sätze eine größere Wahrscheinlichkeit haben extrahiert zu werden. Die Beseitigung dieses Problems wäre aber sehr aufwändig und somit auch sehr kostspielig. Des Weiteren war problematisch, dass die ausgewählten Sätze Verweise auf nicht-ausgewählte Sätze enthielten und daher die Kohärenz der gesamten Zusammenfassung gestört wurde. Als weiteren Nachteil nannten Barzilay und Elhadad das Fehlen von Kontrollmöglichkeiten, um so beispielsweise die Länge der Zusammenfassung zu beeinflussen.

2.4. Aktuelle Forschungsarbeiten auf dem Gebiet der Automatischen Textzusammenfassung

Auf dem Gebiet der Automatischen Textzusammenfassung wird auch heute noch sehr viel Forschung betrieben. Die jährlich stattfindende Document Understandig Conference (DUC), hat sich deshalb die folgenden Ziele gesetzt:

- Überblick über das Feld der Automatischen Textzusammenfassung und dessen aktuelle Forschungsergebnisse
- Evaluation von Zusammenfassungssystemen
- Forschern die Teilnahme an umfangreichen Experimenten ermöglichen.

Nun sollen einige aktuelle Verfahren und Algorithmen zur Automatischen Textzusammenfassung vorgestellt werden, welche teilweise in unserem Zusammenfassungssystem implementiert werden.

2.4.1. Erstellung von Textextrakten mit Hilfe Genetischer Algorithmen (2003)

Enrique Alfonseca und Pilar Rodríguez entwickelten an der Universität von Madrid eine Prozedur zur Erstellung von Zusammenfassungen von einzelnen Dokumenten, welche Sätze extrahiert nach einer neuen, auf genetischen Algorithmen basierenden Zusammenfassungsmethode.

2.4.1.1. Beschreibung der Satzauswahlprozedur

Als erstes erstellt der Algorithmus einige Zusammenfassungen durch die zufällige Auswahl von einer vorgegebenen Anzahl von Sätzen. Jede Zusammenfassung dieser Menge wird als Individuum in einer Population betrachtet und die ausgewählten Sätze einer jeden Zusammenfassung entsprechen dem Genotyp des Individuums, d.h. dem individuellen Satz von Genen.
Es existiert eine Fitnessfunktion, welche auf Heuristiken aus statischen Extraktionsmethoden basiert, die die informativste Zusammenfassung bestimmen soll. Die Fitnessfunktion ist eine Linearkombination folgender Glieder:

- L(S): Summe der Satzlängen aller Sätze einer Zusammenfassung
- W(S): Summe der Positionswerte aller Sätze einer Zusammenfassung, wobei davon ausgegangen wird, dass Sätze am Anfang eines Absatzes informativer sind als die am Ende
- O(S): Wert, der angibt ob die Sätze in der Zusammenfassung in der gleichen Reihenfolge vorkommen wie im Originaltext(O(S)=1) oder nicht (O(S)=0)
- C(S): Wert, der anzeigt, ob die Zusammenfassung Sätze aus allen Paragraphen enthält oder nicht
- P(S): Wert, der beschreibt, ob eine Zusammenfassung die Interessen des Nutzers trifft, welche durch eine Sammlung von für den Nutzer interessanten Dokumenten angegeben sind
- Q(S): wenn die Zusammenfassung auf eine gemachte Anfrage reagieren soll, ist dies ein Wert dafür, ob die Zusammenfassung als Antwort auf die Anfrage gilt
- V(S): Wert, der anzeigt, ob eine Zusammenfassung komplette Sätze enthält, das heißt mit Subjekt und Verb, oder nicht
- I(S): Wert, der angibt, ob in der Zusammenfassung viele Fragen enthalten sind, da Fragen als weniger informativ gelten

Für jedes Individuum der Initialpopulation wird nun der Fitnesswert nach der eben beschriebenen Fitnessfunktion berechnet. Die zwei Individuen mit den geringsten Werten „sterben aus“. Die zwei stärksten Individuen, das heißt jene mit den höchsten Werten, vermehren sich. Diese Vorgehensweise entspricht der natürlichen Selektion. Alle „überlebenden“ Individuen können sich durch zufällige Mutationen verändern, das heißt, dass eine zufällige Anzahl von Sätzen geändert werden kann. Die Kinder der Individuen, die sich vermehren durften, bestehen aus einer zufälligen Anzahl an Sätzen von Mutter und Vater.

Diese Veränderungen der Population durch Mutation und Fortpflanzung, welche über eine gewisse Zeit zu einer homogenen Population führen, finden solange statt, bis sich der höchste Fitnesswert über einige Generationen nicht mehr verändert. Dann endet die Evolution und die beste Zusammenfassung ist entstanden.

2.4.1.2. Ergebnisse der Textzusammenfassung mit Genetischen Algorithmen

Um die erstellten Zusammenfassungen bewerten zu können, wurden Zusammenfassungen mit verschiedenen Kompressionsraten und Nutzerinteressen sowohl von dem Zusammenfassungssystem als auch von drei Menschen erstellt und verglichen.
Die Zusammenfassungen der drei Menschen hatten einen Übereinstimmungsgrad von 60% bis 70%. Die Übereinstimmungen bei den vom System generierten und den von den Menschen erstellten Zusammenfassungen lagen zwischen 50 % (Kompressionsrate: 29%) und 67 % (Kompressionsrate: 46 %). Bei der Bewertung der Lesbarkeit mussten 12 Leser die erstellten Zusammenfassungen lesen und mit Punkten von 1 (sehr schlecht) bis 5 (sehr gut) die Lesbarkeit und den Zusammenhang der Zusammenfassung beurteilen. Der durchschnittliche Wert für die Lesbarkeit lag bei 3.42 mit einer Standardabweichung von 2.63, was für große Abweichungen steht.
Diese guten Ergebnisse zeigten, dass es mit Zusammenfassungserstellung basierend auf genetischen Algorithmen möglich ist, mit einfacher Programmierung eine gute Performanz zu erreichen.
Eine mögliche Erweiterung sehen Alfonseca und Rodríguez in der Berücksichtigung der Länge der Zusammenfassung. Dies könnte geschehen, indem ein Feld von boolschen Werten mit einer Länge, die gleich der Anzahl der Sätze des Originaltextes ist, als Genotyp einer Zusammenfassung betrachtet wird. Eine 0 heißt, der entsprechende Satz gehört nicht zur Zusammenfassung, und eine 1 heißt, der entsprechende Satz ist Teil der Zusammenfassung. So kann eine Zusammenfassung durch Evolution entstehen, deren Länge und deren Satzanzahl nicht festgelegt sind.
Die meiste Forschungsarbeit wollen Alfonseca und Rodríguez aber in die Verbesserung der Fitnessfunktion investieren, da sie davon ausgehen, dass eine Linearkombination der verschiedenen Werte nicht das beste Ergebnis liefert. Zukünftig soll auch die Fitnessfunktion dem Prozess der Evolution unterliegen und sich so verändern. Des Weiteren soll der Zusammenfassungsprozess durch linguistisches Post-Processing erweitert werden, so dass

beispielsweise ein Pronomen, dessen Bezugssubstantiv nicht in der Zusammenfassung vorkommt, durch eben dieses ersetzt wird.

2.4.2. Erstellung von Zusammenfassungen durch Satzreduktion (2000)

Hongyan Jing arbeitete an der Universität von Columbia an einem System, welches extrahierte Sätze, die Teil einer Zusammenfassung werden sollen, reduziert, indem er überflüssige Phrasen entfernt. Dadurch werden Zusammenfassungen nicht nur kürzer und auf das Wesentliche beschränkt, auch die Kohärenz verbessert sich, da Phrasen, die möglicherweise zur Zusammenhangslosigkeit führen, entfernt werden.

2.4.2.1. Der Satzreduktionsalgorithmus

Jings Ziel war es Sätze so zu verkürzen, dass eine knappe Form des Satzes entsteht, die jedoch nicht von der Hauptidee des Satzes ablenkt. Um entscheiden zu können, welche Phrasen irrelevant sind und somit weggelassen werden können, benutzt das Satzreduktionssystem verschiedene Wissensquellen.

- Corpus: Für Traninigs- und Testzwecke kommt ein Corpus aus Originalsätzen und deren reduzierten Formen, welche von Menschen geschrieben wurden, zum Einsatz. Diese Zusammenfassungen kommen vom freien täglichen Nachrichtenservice „Communications-related headlines“ und enthalten Nachrichten aus dem Bereich der Telekommunikation, aber auch Themen wie Recht, Arbeit und Firmenfusion werden abgedeckt.
- Lexikon: Um die verbindlichen Argumente von Verb-Phrasen identifizieren zu können, kommt ein sehr großes, wieder verwendbares Lexikon zum Einsatz. Dieses Lexikon ist ein Kombination aus COMPLEX Syntactic Dictionary, English Verb Classes and Alternations, WordNet Lexical Database und Brown Corpus mit WordNet Bedeutungen.
- WordNet Lexical Database: Um den Fokus im Kontext zu identifizieren wird diese größte Wörterbuchdatenbank genutzt. Mit ihrer Hilfe können Beziehungen zwischen Worten wie Synonyme, Antonyme oder Kausationen gefunden werden.
- Syntaktischer Parser: Jing nutzte den English Slot Grammar Parser von IBM, um die syntaktische Struktur der extrahierten Sätze sowie die thematischen Rollen von Phrasen

(Subjekt, Objekt) analysieren zu lassen und aus den Ergebnissen einen Sentence Parse Tree aufzubauen. (vgl. [Jing 2000, S. 311])

Mit Hilfe dieser Ressourcen läuft die Satzreduktion in fünf Schritten ab. Als erstes erfolgt das syntaktische Parsen, welches aus dem eingegebenen Satz einen Sentence Parse Tree erzeugt. Da auf Grundlage dieses Baumes am Ende entschieden wird, welche Phrasen weggelassen werden können, werden alle Knoten während der nächsten Schritte mit wichtigen zusätzlichen Information zur syntaktischen oder inhaltlichen Bedeutung versehen. Im zweiten Schritt wird durch das grammatikalische Überprüfen festgestellt, welche Satzteile auf keinen Fall gelöscht werden dürfen, da sie für die grammatikalische Richtigkeit des Satzes gebraucht werden. Im dritten Schritt wird auf Grund von Kontextinformationen beurteilt, welche Satzteile mit dem Hauptthema am stärksten verbunden sind. Das System berechnet für jedes Wort eines Satzes einen Wichtigkeitswert, der auf der Anzahl und dem Typ von Verbindungen zwischen diesem Wort und anderen basiert. Ein Wort erhält eine Verbindung zu einem anderen Wort, wenn es eine Wiederholung ist, eine morphologische Beziehung existiert oder wenn die Worte in WortNet verbunden sind. Aus den Wichtigkeitswerten der einzelnen Worte werden die Wichtigkeitswerte der Wortgruppen addiert. Dieser Wert gibt die Bedeutung einer Phrase im Kontext an. Im vorletzten Schritt der Satzreduktion kommt der Korpus mit Sätzen und ihren von Menschen reduzierten Gegenstücken zum Einsatz. Für diese Sätze wird ein Sentence Parse Tree erstellt, in welchem dann die entfernten Phrasen markiert werden. Auf Grundlage dieser Bäume wird die Wahrscheinlichkeit für das Entfernen eines Teilbaums von seinem Elternknoten berechnet. So kann beispielsweise die Wahrscheinlichkeit dafür berechnet werden, dass ein „when“ – Teilsatz entfernt wird, wenn „give“ das Prädikat des Satzes ist. Diese Wahrscheinlichkeiten werden mit der Bayesschen Regel ermittelt.

$$P(\text{„when-clause is removed“}|\text{“v = give“}) = \frac{P(\text{“v = give”}|\text{„when-clause is removed“})\, P(\text{„when-clause is removed“})}{P(\text{“v = give”})}$$

Um möglichst nah an die Satzreduktionspraxis der Menschen zu kommen, werden noch zwei weitere Wahrscheinlichkeiten berechnet: die Wahrscheinlichkeit, dass eine Phrase reduziert wird, und die Wahrscheinlichkeit, dass eine Phrase unverändert bleibt. Auf Basis der Ergebnisse der vorangegangenen Schritte werden im fünften Schritt die Reduktionsentscheidungen getroffen. Dabei wird der Sentence Parse Tree im Top-Down-Verfahren durchlaufen und für jeden Teilbaum entschieden, ob dieser entfernt oder reduziert wird oder ob er unverändert bleibt. Die Entscheidung der Entfernung einer Phrase wird nur dann getroffen, wenn diese Wortgruppe nicht für die Grammatik des Satzes verbindlich ist,

wenn diese Gruppe einen geringen Wichtigkeitswert hat und wenn sie eine recht hohe Wahrscheinlichkeit hat, von Menschen auch entfernt zu werden.

2.4.2.2. Evaluationsergebnisse

Um die Performanz seines Satzreduktionsalgorithmus bestimmen zu können, definierte Jing eine Erfolgsrate, die sich berechnet aus der Anzahl an übereinstimmenden Entscheidungen zwischen Mensch und System geteilt durch die Gesamtanzahl an von Mensch und System getroffenen Entscheidungen.
Für die Evaluation wurde der Corpus mit 400 Sätzen und deren reduzierten Formen gefüllt. Das Programm wurde dann mit 100 Sätzen getestet, wobei die Five-Fold Validation angewandt wurde. Das heißt, es wurde das Experiment für jeden Satz fünfmal wiederholt.
Die durchschnittliche Erfolgsrate betrug 81,3 %. Bei einer gesonderten Betrachtung der Entfernung von Präpositionsphrasen, Nebensätzen, Infinitiven mit to und Gerundien lag die Erfolgsrate bei 43,2 % (vlg. [Jing 2000, S. 314]). Geringe Erfolgsrate konnten bei der Entfernung von Adjektiven in Substantivphrasen und bei der Entfernung von Adverbien in Sätzen oder Wortgruppen mit Verben erzielt werden. Der Grund dafür war, dass der zu kleine Trainingscorpus, kaum die Abhängigkeiten zwischen Adjektiven und Substantiven festlegen konnte, und dass die anderen Wissensquellen nur wenig Aussagen über das Entfernen oder Nicht-Entfernen von Adjektiven lieferten. Da die Entfernung von Adjektiven und Adverbien die Satzlänge nicht signifikant beeinflusst, entschied sich Jing diese Wortart nicht zu entfernen.
Die durchschnittliche Verkürzung der Sätze betrug 32,7 %, während Menschen die Sätze um 41,8 % verkürzten.
Jing sieht viele Möglichkeiten, wenn sein Satzreduktionssystem in einem Zusammenfassungssystem mit anderen Modulen zusammenarbeitet. Zum einen wäre es möglich, die Satzreduktion an Nutzeranfragen anzupassen, indem das System nicht Phrasen entfernt, die für das Hauptthema des Textes belanglos sind, sondern solche, die für die Nutzeranfrage nicht relevant sind. Eine andere denkbar sinnvolle Anwendung wäre ein Feedback zwischen den interagierenden Modulen, denn so könnte das Satzreduktionsmodul bei einem Satz mit geringem Wichtigkeitswert eine Meldung an das Satzauswahlmodul senden oder das Satzreduktionsmodul bekommt eine Nachricht vom darauf folgenden Satzkombinationsmodul und ändert seine Reduktionsentscheidung dahingehend.

2.4.3. Gezielte Textzusammenfassung mit Stichwortlisten (2001)

Timm Euler von der Universität Dortmund erforschte im Rahmen seiner Diplomarbeit die Erstellung gezielter Extrakte unter der Verwendung von maschinellem Lernen. Das maschinelle Lernen wird dabei zur Gewinnung inhaltlicher Vorgaben genutzt.

2.4.3.1. Gezielte Satzextraktion durch Stichwortlisten

Im von Euler erforschten Verfahren zur Erstellung von gezielten Zusammenfassungen kommen als inhaltliche Vorgabe Stichwortlisten zum Einsatz. Um diese Liste von Stichworten zu erhalten, wurden in einem Trainingskorpus alle die Sätze markiert, die für das gewünschte Thema relevant sind. Mit dieser Menge von Beispielsätzen wird nun gelernt bis man als Ergebnis eine Rangliste von gewichteten Stichwörtern erhält. Bevor die Gewichte gelernt werden können, muss eine Stammformreduktion mit Hilfe eines Lexikons der verschiedenen Wortformen vorgenommen werden, bei der verschiedene Formen eines Wortes auf dessen Stammform normalisiert werden. Um jedes Wort in den markierten Sätzen zu gewichten, wurden unterschiedliche Verfahren getestet:

- *Worthäufigkeit*: Bei diesem einfachen Verfahren wird die Anzahl der Vorkommen des Wortes in den markierten Sätzen durch die Anzahl der Vorkommen in den unmarkierten Sätzen geteilt.
- *Information Gain*: Bei diesem auf dem grundlegenden Maß der Entropie basierenden Verfahren erhält ein Wort „sein Gewicht nach der Unterscheidungskraft zwischen positiven und negativen Beispielen (Sätzen)“ [Euler 2001, S. 2]
- *G^2-Statistik*: Etwas abgewandelt von der ursprünglichen Berechnung von G^2 wird ermittelt, wie viel größer die Häufigkeit jedes Wortes in den markierten Beispielsätzen ist im Vergleich zur aus der Häufigkeit in den unmarkierten Sätzen erwarteten Häufigkeit.
- *SVM-Gewichte*: Die Trainingsbeispiele liegen als Vektor in einem mehrdimensionalen Vektorraum vor. Jede Stelle des Vektors steht für ein bestimmtes Wort. Der Wert eines Vektors an einer bestimmten Stelle wird durch das Tf-idf-Maß des Wortes bestimmt. In diesen Vektorraum versucht die Support Vector Machine (SVM) eine Hyperebene so zu legen, dass eine Trennung in positive (markierte) und negative (unmarkierte) Beispielsätze erreicht wird und dass der Abstand zu den nächsten Beispielen so groß wie möglich ist. Die Gleichung dieser Hyperebene liefert Koeffizienten, die als

Gewichte, mit denen jedes Wort für die Lage der Ebene berücksichtigt wurde, angesehen werden können.

Nachdem durch eines dieser Verfahren eine Rangliste von Stichwörtern entstanden ist, erfolgt nun damit die Klassifizierung beliebiger neuer Sätze. Nach der Stammformreduktion der neuen Sätze wird jedem Wort sein Gewicht aus der Stichwortliste zugeordnet, sofern es darin vorkommt. Anschließend werden die Gewichte der Wörter eines Satzes addiert und durch die Anzahl der Worte geteilt. Das Ergebnis daraus ist das Gewicht des Satzes. Nun kann auf dem Trainingskorpus ein Schwellwert ermittelt werden, über welchem das Gewicht eines Satzes liegen muss, damit er markiert wird.

- hoher Schwellwert: Nur wenige Sätze werden markiert, welche aber mit größerer Sicherheit auch wirklich für das Thema relevant sind.
- niedriger Schwellwert: Es werden mehr Sätze ausgewählt, wodurch weniger wirklich relevante Sätze verpasst werden, wodurch aber auch mit höherer Wahrscheinlichkeit einige nicht-relevante Sätze markiert werden.

Um diesen Sachverhalten bewerten zu können und den besten Schwellwert ermitteln zu können, berechnete Euler für jeden Schwellwerttest auf der Trainingsmenge Recall und Precision.

Recall: Anteil der themenrelevanten Sätze, die durch das Verfahren positiv klassifiziert wurden.

Precision: Anteil an positiv klassifizierten Sätzen, die tatsächlich themenrelevant sind.

2.4.3.2. Evaluationsergebnisse

Zur Evaluation des oben beschriebenen Satzextraktionsverfahren simulierte Euler einen Email-to-SMS-Service, bei dem Nutzer Emails, die mit Terminabsprachen zu tun haben, als Kurzform per SMS auf ihr Handy geschickt bekommen. Es wurde ein Korpus mit 560 deutschsprachigen Emails eingesetzt, von denen 50 % Terminabsprachen enthielten, aber nicht ausschließlich Terminabsprachen. Ungefähr 13 % der Sätze des gesamten Korpus wurden als positiv (d.h. terminbezogen) markiert. Bei jedem Test wurden 90 % des Korpus zum Training eingesetzt und der Rest zum Testen.

Verfahen	Recall	Precision	F-Maß	Fallout	Recall	Precision	F-Maß	Fallout
Worthäufigkeit	83,5	79,2	81,2	2,3	82,9	71,7	76,5	3,4
Information Gain	70,5	69,3	69,6	3,6	80,4	71,8	75,6	4,6
G^2-Statistik	73,3	71,5	72,3	3,1	74,9	76,8	57,7	4,7
SVM-Gewichte	79,9	71,2	75,1	5,1	57,5	85,5	68,5	1,1

Abbildung 9: Ergebnisse der Evaluation in % [Euler 2001, S. 3]

Für jedes Gewichtsverfahren wurde das Experiment zehnmal wiederholt, so dass man in der linken Hälfte der obigen Tabelle die zehnfach kreuzvalidierten Recall- und Precisionwerte ablesen kann. Der Fallout-Wert stellt den Anteil der negativen Sätze dar, die fälschlich als positiv markiert wurden. Das F-Maß, welches einen direkten Vergleich ermöglichen soll, berechnet sich aus dem doppelten Produkt von Recall und Precision dividiert durch die Summe aus Recall und Precision. Es konnten folgende Standardabweichung gemessen werden:

- für Recall und Precision: zwischen 4 % und 7 %
- für Fallout: unter 1 %

Wie in der Tabelle abzulesen, erreichten das einfache Verfahren der Worthäufigkeiten sowie das komplexe Verfahren der SVM-Gewichte die besten Ergebnisse. Allerdings erzielten auch die anderen Verfahren brauchbare Ergebnisse. Dabei betrugen die Trainingszeiten für das Lernen der Stichwortlisten wenige Stunden beim Verfahren der SVM-Gewichte und beim Verfahren der Worthäufigkeiten war die Laufzeit vernachlässigbar. Im rechten Teil der Tabelle sieht man die Ergebnisse für das gezielte Zusammenfassen, wenn statt einzelner Sätze immer ganze Texte als positiv bzw. negativ klassifiziert werden. Dies erfordert wesentlich weniger Arbeit und liefert ähnlich gute Resultate.

In einem weiteren Schritt der Evaluation musste eine Reihe von Testpersonen, die entweder die ursprüngliche Email oder die verkürzte SMS kannten, den Email-to-SMS-Service mit einem Fragebogen bewerten. Die Antworten zeigten, dass der Zeitpunkt und Status des Termins am besten erhalten bleiben, dass aber Informationen über den Art des Termins, sowie den Ort und beteiligte Personen oft verloren gehen. Das kann damit erklärt werden, dass diese Wörter nicht nur in terminbezogenen Inhalten auftauchen und deshalb nicht in der Stichwortliste zu finden sind oder aber mit geringen Gewichten. Der Fragebogen zeigte außerdem, dass die Lesbarkeit durch fehlendes Hintergrundwissen litt.

In der Zukunft will Euler dieses Verfahren auf andere Domänen und Texte in anderen Sprachen anwenden. Des Weiteren sieht er Möglichkeiten in der parallelen Anwendung mehrerer Domänen.

3. Implementierung verschiedener Extraktionsalgorithmen

Als Programmieraufgabe dieser Arbeit sollte ein Java-Programm entstehen mit dem die automatische Zusammenfassung von verschiedenen Texten in unterschiedlichen Formaten und mit unterschiedlichen Algorithmen möglich sein sollte. Die Zusammenfassung wird im HTML-Format zurückgegeben. Außerdem sollte das Programm möglichst flexibel sein, das heißt es sollten Zusammenfassungen mit unterschiedlicher Länge von unterschiedlichen Texttypen (HTML, RTF, TXT) mit unterschiedlicher Parametrisierung (abhängig vom gewählten Extraktionsalgorithmus) generiert werden können.

3.1. Analyse der vom Programm zu erledigenden Aufgaben

Bis aus dem Originaltext die Zusammenfassung im HTML-Format entstehen kann, muss das Programm verschiedene Teilaufgaben erfüllen:

- Einlesen der Charakteristika für die Zusammenfassung aus der grafischen Nutzeroberfläche
- Umwandeln des Originaltextes in ein Objekt der Klasse Text
- mit Hilfe des gewählten Zusammenfassungsalgorithmus die Sätze für die Zusammenfassung bestimmen
- Konstruktion, Speicherung und Ausgabe der Zusammenfassung auf der GUI

3.1.1. Grafische Nutzeroberfläche

Auf der GUI können alle nötigen Einstellungen für die Zusammenfassung getroffen werden.

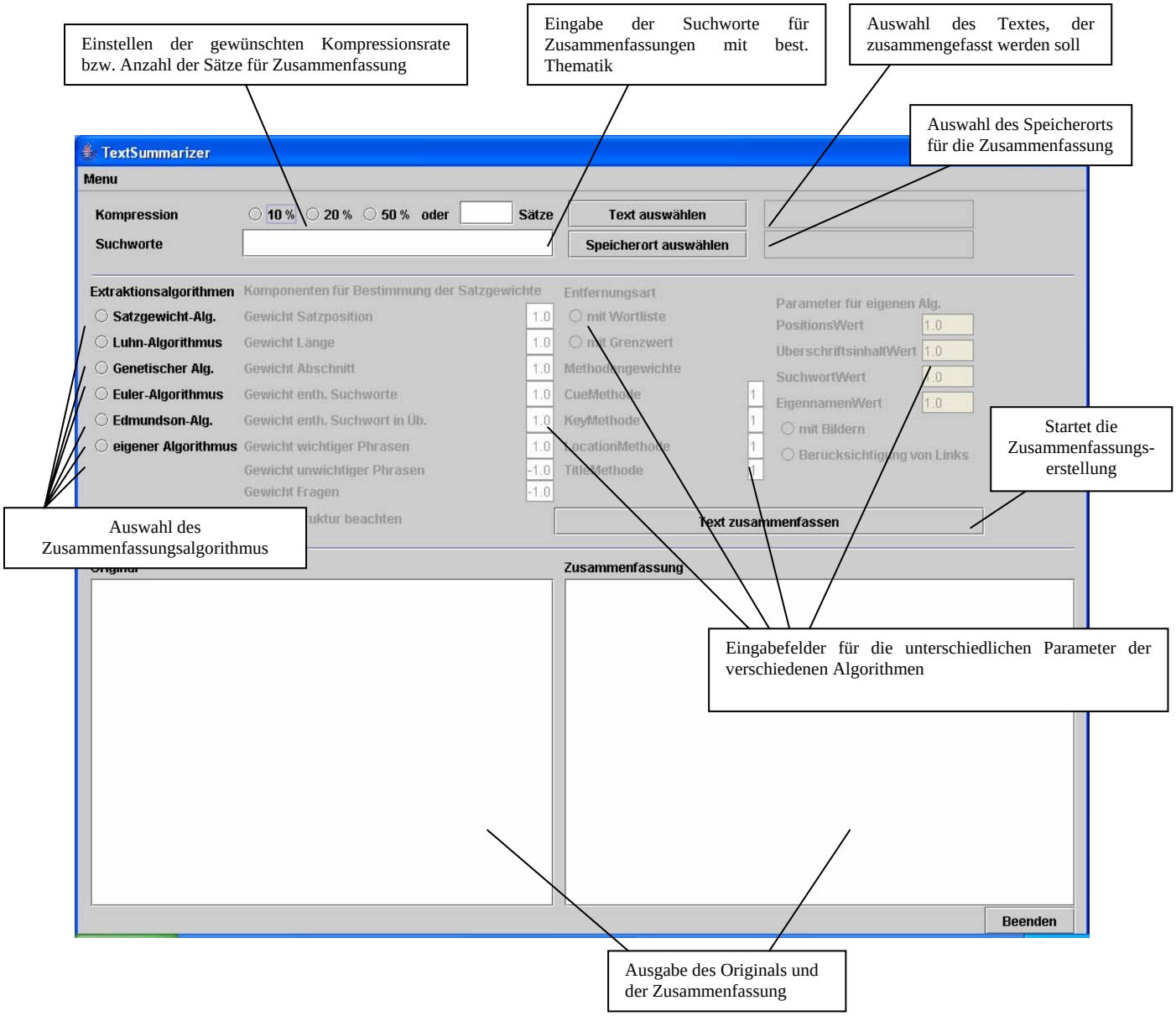


Abbildung 10: Screenshot unseres Textzusammenfassungsprogramms

3.1.2. Einlesen des Originaltextes und Anlegen eines Text-Objektes

Um aus dem gewählten Text eine Zusammenfassung erstellen zu können, muss er in eine Form gebracht werden, welche einen schnellen Zugriff auf verschiedene Textelemente (Satz, Abschnitt, Überschrift, Bilder, …) und Textcharakteristika (Anzahl Sätze, Anzahl Abschnitte, Speicherort, …) bietet. Zu diesem Zweck wurde die Klasse Text implementiert. Diese Klasse nutzt zur Speicherung des Textinhalts die Klassen Abschnitt und Satz (siehe Abbildung 11).

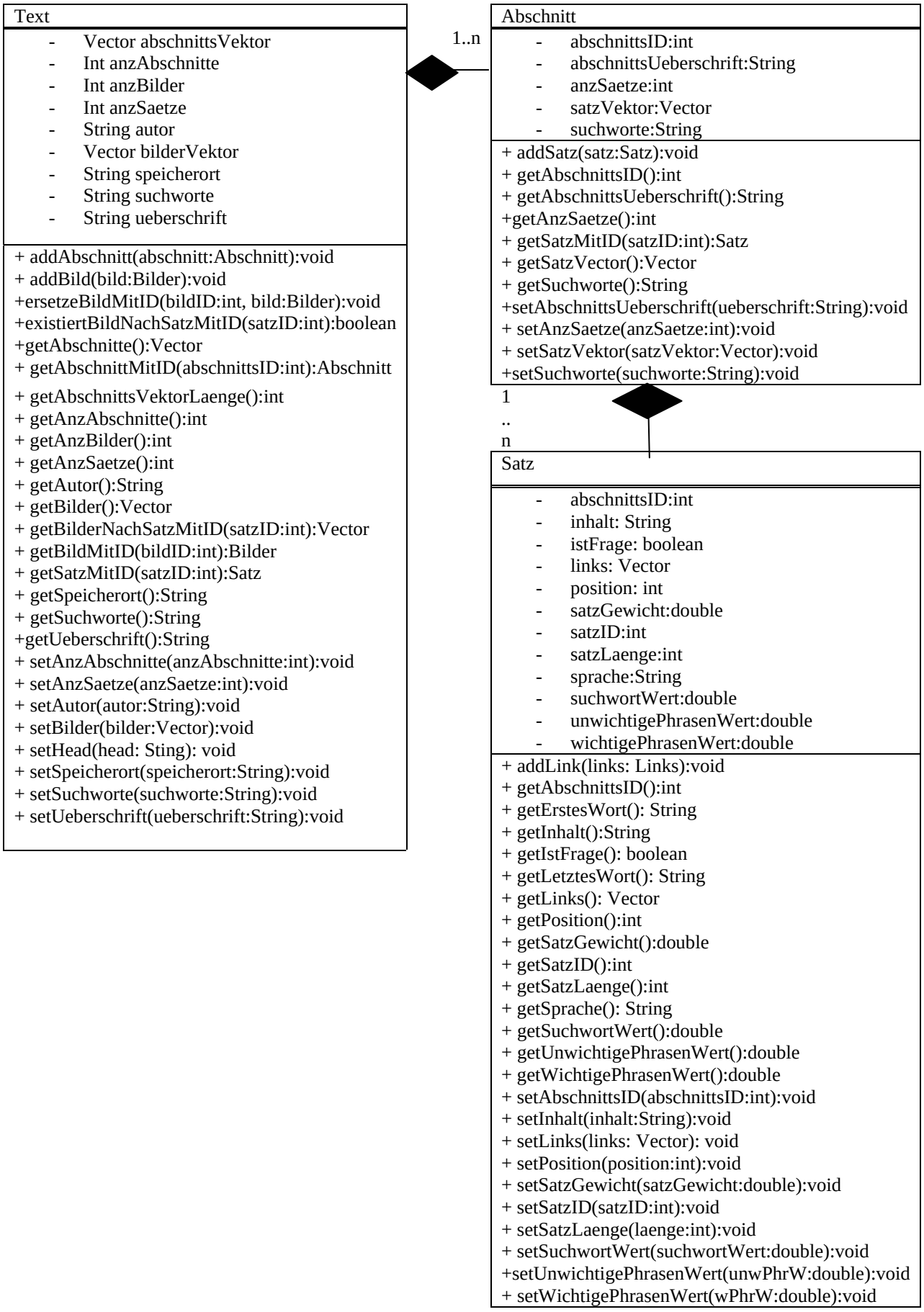


Abbildung 11: Klassendiagramm der Klassen, die für die Speicherung eines Textes benötigt werden

Da das Programm Dokumente in verschiedenen Dateiformaten verarbeiten soll, gibt es unterschiedliche Funktionen zur Erstellung eines Text-Objektes. Diese Funktionen sind Methoden der Klasse TextZusammenfassen, die neben allen Methoden zur automatischen Zusammenfassungsgeneration auch die Attribute zur Parametrisierung der Zusammenfassung beinhaltet.

TextZusammenfassen
- anzSätzeFürZusammenfassung: int - ausgewaehlteBilder: Vector - cueDatei: File - edmundsonParameter:ParameterEdmundson - entfernungsArt:int - gesamerText:Text - gewichte:Gewichte - headingDatei: File - inhaltDerZusammenfassung: String - meineExtraktionsParameter: MeineExtraktionsParameter - mitBildern:boolean - mitLinks: boolean - nullDatei: File - population:int[][] - speicherortOriginal:File - speicherortZusammenfassung:File - stichwortDatei: File - suchworte:String - textstrukturbeachten:boolean - wortliste:Vector - zusammenfassungsAlgorithmus:int - zusammenfassungsGrad:double - zusammenfassungsGroesse:double
- berechneCueGewicht(text:Text,satzID:int):double - berechneHäufigkeit(inhalt: String, wort: String): int - berechneKeyGewicht(text:Text,satzID:int,keyGlossar:Vector):double - berechneLocationGewicht(text:Text,satzID:int):double - berechneTitleGewicht(text:Text,satzID:int):double - bestimmeMax(feld:double[ ]):int - bestimmeMin(feld:double[ ]):int - bestimmeSatzGewicht(satz:Satz,text:Text,gewichte:Gewichte,textstrukturbeachten:boolean):double - bestimmeWortHaeufigkeiten(text:Text):Vector - bestimmeZweitMax(feld:double[]):int - bestimmeZweitMin(feld:double[]):int - bildIstNichtWichtig(bildID: int, ausgewählterBilder: Vector): boolean - crossoverOperator(mutter:int[],vater:int[],zufall:Random):int[] - edmundsonAlgorithmus(text:Text, parameter:ParameterEdmundson):int[] - entferneAllgemeineWorte(worte:Vector,entfernungsArt:int):Vector - erhoeheHaufigkeitVonWortID(ID:int,worte:Vector):Vector - erstelleTextAusHTML(inhalt:String, speicherort:String):Text - erstelleTextAusRTF(inhalt:String, speicherort:String):Text - erstelleZusammenfassungInHTML(population:int[],text:Text,speicherort:String,bilderGewaehlt:boolean):void - eulerAlgorithmus(text: Text): int[] - evolution(population:int[],text:Text,gewichte:Gewichte):void - existierenDoppelteSaetze(population:int[]):int - fassezusammen():boolean - fitnessFunction(population:int[],text:Text,gewichte:Gewichte):double - getAnzahlSätzeFürZusammenfassung(): int - getCueDatei(): File - getHeadingDatei(): File

- getInhaltDerZusammenfassung(): String
- getMeineExtraktionsParameter(): MeineExtraktionsParameter
- getNullDatei(): File
- getStichwortDatei(): File
- getSuchworte(): String
- getWortIDMitWort(gesuchtesWort:String,worte:Vector):int
- getWortMitID(ID:int,worte:Vector):Wort
- getZusammenfassungsAlgorithmus(): int
- initPopulations(anzPopulationen:int):int[][]
- initTextZusammenfassen(): boolean
- initWortliste(datei: String): void
- istEintragInDictionaryEnthalten(wort:String,dictionary:Vector): boolean
- istWortInSatzEnthalten(satz:String,wort:String): boolean
- kommtVorInWortliste(wort:String): boolean
- ladeDictionary(datei:String): Vector
- ladeDictionaryOhneGewichte(datei:String):Vector
- ladeEulerWortliste(datei: String): Vector
- ladeKeyGlossar(text:Text): Vector
- ladeTitleGlossar(text:Text): Vector
- legeInitCueDateiAn(): String
- legeInitDateiAn():void
- legeInitHeadingDateiAn(): String
- legeInitNullDateiAn(): String
- legeStichwortDateiAn(): String
- luhnAlgorithmus(text:Text, entfernungsArt:int):int[]
- meinExtraktionsalgorithmus(text: Text, meineExtraktionsParameter: MeineExtraktionsParameter): int[]
- mutationsOperator(population:int[],groessteSatzID:int):int[]
- ordneNachSignifikanzAbsteigend(zerlegteSaetze:SatzZerlegung[]):SatzZerlegung[]
- ordnePopulationAufsteigend(population:int[]):int[]
- sentenceRanking(text:Text, gewichte:Gewichte,textstrukturBeachten:boolean):int[]
- setAnzSätzeFürZusammenfassung(anzSätzeFürZusammenfassung: int): void
- setCueDatei(cueDatei: File): void
- setEdmundsonParameter(edmundsonParameter:ParameterEdmundson): void
- setEntfernungsArt(entfernungsArt:boolean):void
- setGesamterText(gesamterText:Text):void
- setGewichte(gewichte:Gewichte):void
- setHeadingDatei(headingDatei: File): void
- setMeineExtraktionsParameter(meineExtraktionsParameter: MeineExtraktionsParameter): void
- setMitBildern(mitBildern:boolean):void
- setMitLinks(mitLinks: boolean): void
- setNullDatei(nullDatei: File): void
- setSpeicherortOriginal(speicherortOriginal:File):void
- setSpeicherortZusammenfassung(speicherortZusammenfassung:File):void
- setStichwortDatei(stichwortDatei: File): void
- setSuchworte(suchworte:String):void
- setTextstrukturBeachten(textstruktureBeachten:boolean):void
- setZusammenfassungsAlgorithmus(zusammenfassungsAlgorithmus:int):void
- setZusammenfassungsGrad(kompression: double): void
- setZusammenfassungsGroesse(anzSaetzeImText:int):void
- suchworteEnthalten(inhalt:String,suchworte:String):boolean
- vergleicheMitStichworten(wort: String, wortliste: Vector): double

Abbildung 12: Klassendiagramm der Klasse TextZusammenfassen

3.1.2.1.Einlesen von HTML-Dokumenten

Das Zusammenfassen von Dokumenten im HTML-Format, also von Webseiten, birgt einige Schwierigkeiten in sich, da im Text auch Information zum Layout, Werbeeinblendungen,

Navigationselemente und ähnliches enthalten sein können. Andererseits können einige dieser Informationen beim Finden von Überschriften oder Absätzen sehr hilfreich sein.

Die Funktion erstelleTextAusHTML, welche ein Text-Objekt zurückgibt, erhält als Eingabe den Speicherort, an dem die HTML-Datei zu finden ist, außerdem wird der Inhalt der Datei als String übergeben.
Zuerst wird nach dem Titel- und dem Autor-Tag gesucht, um so den Titel und den Autor der HTML-Datei herauszufinden und im Text-Objekt zu speichern. Danach wird der Header des Originals für die Zusammenfassung gespeichert. Anschließend wird der Inhalt der Datei an den <h>-Tags, welche den Anfang einer Überschrift kennzeichnen, in viele kleinere Strings, welche jeweils einen Absatz enthalten, aufgesplittet. Die Anzahl der entstandenen Teilstrings wird als die Abschnittanzahl im Textobjekt gespeichert. In einer Schleife wird für jeden dieser Absatz-Strings ein Objekt der Abschnitt-Klasse erzeugt, welches eine eindeutige Abschnitts-Id erhält. Dann wird die Abschnittsüberschrift extrahiert und gespeichert. Danach wird der Abschnitts-String an den Satzzeichen „.“, „?“ und „!“ in die einzelnen Sätze aufgespaltet. In einer inneren Schleife wird für jeden so entstandenen Satz-String des Abschnitts ein Objekt der Satz-Klasse angelegt. Dieses Objekt erhält eine eindeutige ID, den Satz-String als Inhalt des Satzes, die Satzlänge, die Position im Abschnitt, die Abschnitts-Id und die Suchworte. Im Konstruktor der Satz-Klasse, der aufgerufen wurde, wird dann automatisch der Suchwortwert und die Werte für die enthalten Hinweisphrasen berechnet. Sollte ein Satz allerdings kürzer als 5 Zeichen und nicht der erste eines Abschnitts sein, wird davon ausgegangen, dass der Punkt, an dem der String aufgeteilt wurde, kein Satzpunkt war, sondern eine Abkürzung beendet. In diesem Fall wird der Inhalt an den vorherigen Satz angehangen.
Sollte im Satz-String das <IMG>-Tag enthalten sein, wird kein Satz-Objekt sondern ein Bilder-Objekt erzeugt (siehe Abbildung 13). In diesem Objekt werden eine BildID, die ID des vorherigen Satzes, die Abschnitts-Id und der Inhalt des Satz-Strings gespeichert. Sollte der Satz danach mit Worten wie „Bild“ oder „Abbildung“ beginnen, so wird dieser String nicht als Satzobjekt abgespeichert, sondern für das Bildobjekt als Bildunterschrift gespeichert.
Bevor ein Abschnitt an den Abschnittsvektor des Textobjektes angehangen wird, wird überprüft, ob der Abschnitt eine Überschrift hat. Ist dies nicht der Fall, wird die Überschrift des vorherigen Abschnitts eingefügt.

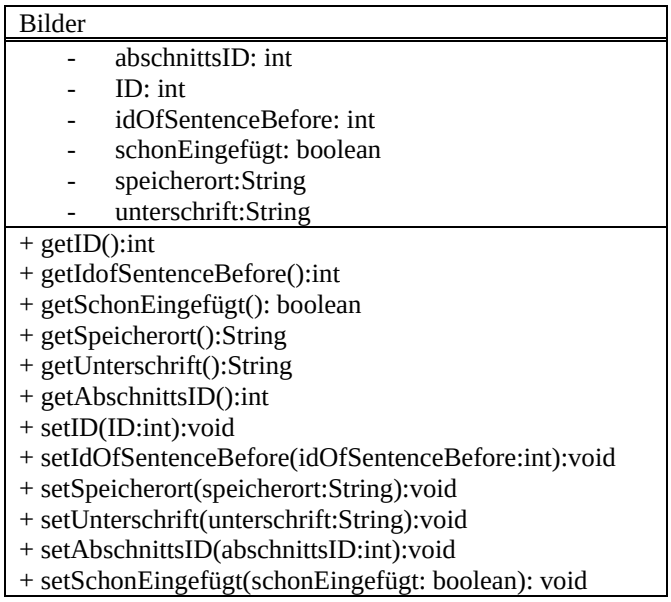


Abbildung 13: Klassendiagramm der Klasse Bilder

3.1.2.2.Einlesen von RTF-Dokumenten

Das größte Problem beim Verarbeiten von Dokumenten im RTF-Format sind die enthaltenen Formatanweisungen, welche umständlich entschlüsselt werden müssten. Da diese Informationen für die Zusammenfassung nicht benötigt werden, umgingen wir eine umständlich Konvertierung, indem der Inhalt aus dem Java EditorPane einfach über die getText()-Methode eingelesen wird. Der so erhaltene Inhaltsstring wird zusammen mit dem Speicherort an die Methode erstelleTextAusRTF() übergeben, welche daraus ein Textobjekt erstellt. Da bei diesem Dateiformat nicht auf Tags, die den Anfang und das Ende eines Abschnitts oder eine Überschrift kennzeichnen, zurückgegriffen werden kann, legten wir fest, dass immer dann ein neuer Abschnitt beginnt, wenn zuvor ein NewLine-Zeichen („\n“) erkannt wurde. Allerdings stellten wir fest, dass dieses Zeichen auch nach einer Überschrift und somit vor einem eigentlichen Abschnitt zu finden ist. Deshalb musste überprüft werden, ob es sich um eine Überschrift oder den eigentlichen Abschnitt handelt.
Um die Überschrift eines Abschnittes zu finden, wurden zwei Kriterien eingesetzt:

- *Kein Satzpunkt*: Kommt in einem Abschnitt der durch zwei NewLine-Zeichen begrenzt wird, keine Satzpunkt vor, so kann davon ausgegangen werden, dass es sich um eine Überschrift handelt. Als problematisch erwies sich allerdings, dass bei RTF-Dokumenten mit Bildern auch die Bildunterschriften zwischen zwei NewLine-Zeichen liegen und somit irrtümlich als Überschriften erkannt werden

- *Beginnt mit Aufzählung*: Beginnt ein Abschnitt mit einem Aufzählungsstring, so wird von einer Überschrift ausgegangen.

Wurde eine Überschrift gefunden, wird in einem Flag die Überschrift gespeichert, so dass sie, wenn der dazugehörige Abschnitt entdeckt wird, als Abschnittsüberschrift zu diesem Abschnitt hinterlegt werden kann.
Wurden in dem potenziellen Abschnitt keine Überschriftskriterien aufgedeckt, so wird der Abschnitt an den Satzzeichen „.“, „?“ und „!“ in einzelne Satzstrings zerlegt. Für jeden Satzstring wird ein Satzobjekt mit Satz-Id, Position im Abschnitt, Satzlänge und den restlichen Satzattributen erzeugt und an den Satzvektor des Abschnitts angehangen. Eine Ausnahme davon bilden Sätze, deren Nachfolgesatz nicht mit einem Großbuchstaben oder einer Zahl beginnt. Wenn dies der Fall ist kann davon ausgegangen werden, dass der Nachfolgesatz ein Teil des vorherigen Satzes ist. Für diese Sätze wird kein Satz-Objekt erzeugt, sondern ihr Inhalt wird mit dem Satzinhalt des Nachfolgesatzes zusammengefügt. Wurde ein Abschnitt ohne Überschrift angelegt, so gingen wir davon aus, dass er zur selben Überschrift wie der vorherige Abschnitt gehört, so dass die Abschnittsüberschrift des Vorgängerabschnitts auch für diesen Abschnitt als Überschrift gespeichert wird.

3.1.2.3.Einlesen von TXT-Dokumenten

Das Einlesen von TXT-Dateien geschieht auch mit der erstelleTextAusRTF()-Methode, da der TXT-Text in der gleichen Art und Weise vorliegt wie der RTF-Text, welcher aus dem EditorPanel eingelesen wurde. Allerdings muss bevor die Methode aufgerufen wird, der Text über die dateiEinlesenMitNL()-Methode aus der Datei eingelesen werden. Diese Methode gibt den Inhalt der Datei als String zurück. Dieser String wird zusammen mit dem Speicherort dann an die erstelleTextAusRTF()-Methode übergeben, welche den Text als Textobjekt zurückgibt.

3.1.3. Auswahl der relevanten Sätze für die Zusammenfassung anhand verschiedener Algorithmen

Zur Selektion der für die Zusammenfassung relevanten Sätze setzten wir zum Vergleich verschiedene Extraktionsalgorithmen ein. Alle diese Algorithmen erzeugen als Rückgabeelement ein Integer-Feld, in welchem die Satz-Ids der relevanten Sätze gespeichert sind. Dieses Feld erhält im Anschluss eine Prozedur als Eingabe, die daraus eine HTML-Zusammenfassung konstruiert.

3.1.3.1. Genetischer Satzextraktionsalgorithmus

Dieser Extraktionsalgorithmus entspricht im Allgemeinen der Satzselektionsidee von Enrique Alfonseca und Pilar Rodríguez (vgl. Kapitel 2.4.1.).
Wie bereits beschrieben, wird bei diesem Algorithmus mit einer zufällig initialisierten Anfangspopulation, welche aus einer Menge von Zusammenfassungsindividuen besteht, begonnen. Dabei versteht man unter einem Zusammenfassungsindividuum eine bestimmte Anzahl an Satz-Ids, wobei jede Satz-Id in einem Individuum nur einmal vorkommen darf. Die Anzahl wird dabei von der Textgröße und der Kompressionsrate vorgegeben.

[1, 3, 5, 8, 9]; [0, 2, 4,8, 9]; [2, 3, 4, 7, 8]

Abbildung 14: Mögliche Zusammenfassungsindividuen eines Textes mit 10 Sätzen bei einer Kompressionsrate von 0,5

Diese Population wird als ein Feld von Integer-Feldern der gleichen Größe repräsentiert. Mit Hinblick auf die Genetik entsprechen die Satz-Ids dem Genom und damit dem Satz aller Gene eines Individuums. Wie gut ein Individuum als Zusammenfassung geeignet ist, besagt sein Fitnesswert.
Wie in der Natur ist auch die Zusammenfassungspopulation den Gegebenheiten der Evolution ausgesetzt. Das heißt, sie verändert sich durch Mutation und Vererbung.

3.1.3.1.1. Berechnung des Fitnesswertes

Der Fitnesswert eines Zusammenfassungsindividuums wird über die Methode fitnessFunction() der TextZusammenfassen-Klasse berechnet. Als Eingabeparameter werden die Individuen also die Zusammenfassungen, der Text und die Gewichte für die einzelnen Merkmale übergeben. Diese Gewichte dienen als Gewichtung der einzelnen Komponenten in der Linearkombination und geben so an wie relevant die einzelnen Komponenten bei der Bestimmung des Fitnesswertes sind.

$$\begin{aligned}\text{FitnessWert} = {} & \text{WichtigPhrasenWert} * \text{Gewicht}_{\text{WichtigPhrasen}} + \\ & \text{UnwichtigePhrasenWert} * \text{Gewicht}_{\text{UnwichtigPhrasen}} + \\ & \text{ZusammenfassungslängeWert} * \text{Gewicht}_{\text{Zusammenfassungslänge}} + \\ & \text{PositionsWert} * \text{Gewicht}_{\text{Satzposition}} + \\ & \text{AbschnittsWert} * \text{Gewicht}_{\text{Abschnitt}} + \\ & \text{SuchwortWert} * \text{Gewicht}_{\text{Suchworte}} + \\ & \text{SuchworteInÜberschriftWert} * \text{Gewicht}_{\text{SuchworteInÜberschrift}} + \\ & \text{FrageWert} * \text{Gewicht}_{\text{Fragewert}}\end{aligned}$$

Abbildung 15: Berechnungsvorschrift für den Fitnesswert

3.1.3.1.1.1. Berechnung der einzelnen Komponenten

WichtigePhrasenWert:
Die Grundlage für diesen Wert ist die Annahme, dass relevante Sätze Worte enthalten, die auf die Wichtigkeit des Satzes hindeuten, wie zum Beispiel „zusammenfassend“, „wichtig“ oder „signifikant“. Diese Worte sind in der Cue.txt-Datei hinterlegt und mit positiven Gewichten gekennzeichnet. Bei dem Anlegen eines Satzobjektes wird überprüft, wie viele dieser auf die Wichtigkeit des Satzes hinweisenden Worte in dem Satz enthalten sind. Diese Anzahl wird durch die Anzahl aller in der Datei enthaltenen Hinweis-Worte geteilt. Dieser Wert ist der WichtigePhrasenWert eines Satzes. Um den WichtigePhrasenWert einer Population zu ermitteln, werden die Werte der darin enthaltenen Sätze addiert.

UnwichtigePhrasenWert:
Basierend auf der Meinung das nicht-relevante Sätze an bestimmten Worten wie „kaum“, „unwichtig“ oder „bedeutungslos“ erkannt werden können, sind auch diese in der Cue.txt-Datei mit negativen Gewichten gespeichert. Der UnwichtigePhrasenWert für jeden Satz und für jede Population wird ähnlich dem WichtigePhrasenWert berechnet.

ZusammenfassungslängeWert:
Ausgehend von der Annahme, dass längere Sätze besser für eine Zusammenfassung sind als kurze, wird für jeden Satz die Länge, das heißt die Wortanzahl, ermittelt und für den ZusammenfassungslängeWert einer Population die Werte der einzelnen Sätze aufsummiert.

PositionsWert:
Da laut Alfonseca und Rodríguez Zusammenfassungen, die Sätze vom Anfang eines Abschnitts enthalten, besser sind als die mit Sätzen vom Ende, wird für jeden Satz der

Positionswert folgendermaßen berechnet: PositionsWert = 1 / Position im Abschnitt. Auch hier werden die Werte der einzelnen Sätze der Population addiert um den PositionsWert für die Zusammenfassungspopulation zu erhalten.

AbschnittsWert:
Ausgehend von der Hypothese, dass Zusammenfassungen zu bevorzugen sind, die Sätze aus möglichst allen Abschnitten enthalten, wird für jede Population ein AbschnittsWert berechnet. Dafür wird für die Zusammenfassungspopulation überprüft, aus welchen Abschnitten die einzelnen Sätze stammen. Der AbschnittsWert berechnet sich dann aus der Anzahl der Abschnitte, die mit mindestens einem Satz in der Zusammenfassung vertreten sind, geteilt durch die Gesamtanzahl an Abschnitten.

SuchwortWert:
Dieser Wert und der SuchwortInÜberschriftWert ermöglichen die Anpassung der Zusammenfassung an die Nutzerinteressen, indem die über die GUI eingegebenen Suchworte bei der Textzusammenfassung berücksichtigt werden. Der SuchwortWert eines Satz gibt das Verhältnis zwischen der Anzahl im Satz enthaltenen Suchworte zur Gesamtanzahl an übergebenen Suchworten an. Der SuchwortWert der Population entspricht der Summe der SuchwortWerte der Einzelsätze.

SuchwortInÜberschriftWert:
Bei diesem Wert werden die Abschnittsüberschriften auf Vorkommen der angegebenen Suchworte überprüft. Der SuchwortInÜberschriftWert jedes Satzes eines Abschnitts berechnet sich aus der Anzahl in der Überschrift enthaltener Suchworte geteilt durch die Gesamtanzahl an Suchworten.

FrageWert:
Dieser Wert dient der Eliminierung von Sätzen mit wenigen Informationen. Da angenommen wird, dass der Informationsgehalt von Fragen sehr gering ist, geht der FrageWert mit negativem Gewicht in die Berechnung des Fitnesswertes ein. Der FrageWert einer Population stellt das Verhältnis an Fragen in der Population dar.

OrdnungsWert:
Theoretisch hatten Alfonseca und Rodríguez noch diesen Wert zur Berücksichtigung der Ordnung der Population vorgesehen. Da aber die Populationen sowohl nach der Initialisierung als auch nach jedem Evolutionsdurchlauf nach den Satz-Ids aufsteigend geordnet werden,

sind die Sätze stets in der richtigen Reihenfolge und dieser Wert wäre stets 1. Daher wird er für den Fitnesswert nicht berücksichtigt.

3.1.3.1.2. Evolution der Zusammenfassungspopulation

Nachdem für jedes Individuum der Initialpopulation der Fitnesswert berechnet wurde, beginnt der erste Evolutionsdurchlauf. Die beiden Individuen mit den kleinsten Fitnesswerten „sterben aus“, das heißt, sie werden gelöscht. An ihre Stelle treten die „Nachkommen“ der beiden Individuen mit den größten Fitnesswerten. Im Anschluss daran unterliegen alle jetzt aktuellen Zusammenfassungsindividuen der zufälligen Mutation.

3.1.3.1.2.1. Erzeugung von Nachkommen

Das Erzeugen von so genannten Nachkommen erfolgt mit der Methode crossoverOperator(). Dieser erhält die Mutter- und Vaterzusammenfassung sowie ein Objekt der Klasse Random als Eingabeparameter und liefert die Kindzusammenfassung als Integer-Feld zurück.
Die Kindzusammenfassung ist genauso groß wie die Elternindividuen und erhält eine zufällige Anzahl an Sätzen von der Vaterzusammenfassung und den Rest von der Mutterzusammenfassung. Dabei wird mit der Methode existierenDoppelteSaetze() überprüft, ob kein Satz doppelt vorkommt. Ist dies der Fall wird ermittelt, ob die Sätze aus einem Elternteil stammen oder durch Kombination entstanden sind. Dann werden die doppelten Sätze solang durch zufällige Sätze des betroffenen Elternindividuums ersetzt, bis keine doppelten Sätze mehr existieren. Als Letztes wird die Kindzusammenfassung noch aufsteigend geordnet.

3.1.3.1.2.2. Mutation der Population

Wie in der Natur unterliegen die Zusammenfassungsindividuen der Mutation, so dass neue Individuen mit verändertem Genotyp, d.h. veränderten Satz-Ids, entstehen können. Die Mutation eines Individuums übernimmt der mutationsOperator(), welcher die betroffene Zusammenfassung und die größte Satz-Id des Textes übergeben bekommt. Dabei wird für jedes Zusammenfassungsindividuum mit einem boolschen Zufallswert entschieden, ob es mutiert oder nicht.
Wenn ein Individuum mutiert, wird ein zufälliger Satz dieses Individuums ausgewählt, der durch einen zufälligen anderen Satz ersetzt wird. Auch hier wird darauf geachtet, dass keine doppelten Sätze in dem Zusammenfassungsindividuum enthalten sind. Bevor die so

veränderte Zusammenfassung zurückgegeben wird, wird sie noch nach den Satz-Ids aufsteigend geordnet, da durch den „mutierten“ Satz diese Ordnung gestört sein kann.

3.1.3.1.3. Bestimmen der endgültigen Zusammenfassung

Im Laufe der 50 Evolutionsschritte sollten – ähnlich wie in der Natur – die stärksten Individuen sich durchgesetzt haben, so dass sich die Fitnesswerte der einzelnen Zusammenfassungen immer mehr angleichen. Daher wird nach 50 Evolutionsschritten das Zusammenfassungsindividuum mit dem größten Fitnesswert bestimmt und dieses als für die Zusammenfassung relevante Sätze zurückgegeben.

3.1.3.2. Satzgewichtalgorithmus

Bei diesem Extraktionsalgorithmus erfolgt die Bestimmung relevanter Sätze anhand verschiedener statistischer und linguistischer Kriterien, welche unterschiedlich gewichtet werden können. Die Satzselektion erfolgt mit der sentenceRanking()-Methode, welche den Originaltext als Textobjekt, die gewählte Kompressionsrate, die Gewichte für die einzelnen Komponenten und ein Flag, welches angibt, ob die Textstruktur beachtet werden soll, als Eingabeparameter erhält.

Es werden zuerst alle Sätze des Originaltextes gewichtet und anschließend anhand ihrer Gewichte absteigend geordnet. Dann wird auf Grundlage der Anzahl der Sätze des Originals und der gewählten Kompressionsrate die Größe der Zusammenfassung bestimmt. Danach werden die am höchsten gewichteten Sätze des Originals ausgewählt, so dass eine Zusammenfassung mit der vorher ermittelten Größe entsteht. Die Satz-Ids dieser Sätze werden in einem Integer-Feld aufsteigend sortiert zurückgegeben.

3.1.3.2.1. Berechnen der Satzgewichte

Die Berechnung des Gewichtes eines Satzes erfolgt auf ähnliche Art und Weise wie die Berechnung des Fitnesswertes eines Zusammenfassungsindividuums. Das Satzgewicht ist eine Linearkombination unterschiedlich stark gewichteter Textmerkmalswerte.

$$
\begin{aligned}
\text{Satzgewicht}(s) = \quad & \text{Gewicht}_{\text{Satzlänge}} * \text{LängeWert}(s) + \\
& \text{Gewicht}_{\text{Suchwort}} * \text{SuchwortWert}(s) + \\
& \text{Gewicht}_{\text{Satzposition}} * \text{PositionsWert}(s) + \\
& \text{Gewicht}_{\text{WichtigePhrasen}} * \text{WichtigePhrasenWert}(s) + \\
& \text{Gewicht}_{\text{UnwichtigePhrasen}} * \text{UnwichtigePhrasenWert}(s) + \\
& \text{Gewicht}_{\text{Abschnitt}} * \text{AbschnittsWert}(s) + \\
& \text{Gewicht}_{\text{SuchwortInÜberschrift}} * \text{SuchwortInÜberschrift}(a(s))
\end{aligned}
$$

Abbildung 16: Berechnungsvorschrift für die Bestimmung des Gewichtes des Satzes s aus dem Abschnitt a

Die einzelnen Textmerkmalswerte können stets einen Maximalwert von 1 erreichen, damit eine gezielte Gewichtung einfacher vollzogen werden kann. Das hat sich als sinnvoll erwiesen, da, als anfangs die Satzlänge direkt eingegangen ist, dieser eine Wert bei gleicher Gewichtung aller Merkmale viel größer und damit viel bedeutender gewesen ist.
Die Komponenten *WichtigePhrasenWert*, *UnwichtigePhrasenWert*, *SuchwortWert*, *SuchwortInÜberschriftWert* und *PositionsWert* berechnen sich genau so wie bei der Berechnung des Fitnesswertes (siehe Kapitel 3.1.3.1.1.1.).
Der *AbschnittsWert* bewertete Sätze abhängig von dem Abschnitt, in dem sie stehen. Dabei gilt die Annahme, dass die Abschnitte am Anfang des Textes wichtiger sind als die Abschnitte am Ende. Daher berechnet sich dieser Wert aus 1/Abschnittsposition im Text. Bei wissenschaftlichen Texten hat man allerdings meist auch am Ende eines Textes, also im letzten Abschnitt relevante Sätze, da hier noch einmal die wichtigsten Punkte des Dokumentes zusammengefasst werden. Wenn solch ein Text vorliegt, sollte der Radiobutton „Textstruktur beachten“ gewählt werden, da dann die Sätze des letzten Abschnitts die gleichen Abschnittswerte erhalten wie die Sätze des ersten Abschnitts.
Der *LängeWert* eines Satzes berechnet sich aus der Anzahl der Worte des Satzes (entspricht der Satzlänge) geteilt durch die Satzlänge des längsten Satzes und gibt längeren Sätzen daher mehr Gewicht als kurzen.

3.1.3.2.2. Selektion relevanter Sätze

Die relevanten Sätze eines Textes zeichnen sich durch hohe Satzgewichte aus, da dies darauf hinweist, dass viele Textmerkmale, die auf die Wichtigkeit eines Satzes deuten, in dem betreffenden Satz gefunden wurden. Daher werden die Sätze nach ihren Gewichten absteigend geordnet. Da unser Programm flexibel für unterschiedliche Zusammenfassungsgrößen (10%, 20% und 50% der Originallänge bzw. direkt eingegebene

Satzanzahl) verwendet werden kann, werden nicht starr die 10 oder 20 höchsten Sätze ausgewählt, sondern die Anzahl der Sätze kann entweder direkt in der GUI eingegeben werden oder errechnet sich aus der Kompressionsrate multipliziert mit der Anzahl an Sätzen im Originaltext. Die so ausgewählten Sätze werden noch nach der Satz-Id aufsteigend sortiert, damit sie in der gleichen Reihenfolge wie im Ausgangstext vorliegen, und als Integer-Feld von Satz-Ids zurückgegeben.

3.1.3.3. Extraktionsalgorithmus nach Luhn

Der dritte Extraktionsalgorithmus orientiert sich an dem Algorithmus von Luhn aus dem Jahre 1958. Hier werden nur statistische Werte berücksichtigt. Die Methode luhnAlgorithmus() erhält den Text und einen Integerwert, der angibt, wie allgemeine Worte entfernt werden sollen, als Input. Auch dieser Extraktionsalgorithmus liefert die relevanten Sätze als Satz-Ids in einem Integer-Feld zurück.
Der Algorithmus läuft in vier Schritten ab:

1. Ermitteln der Worthäufigkeiten
2. Entfernen zu häufiger und zu seltener Worte
3. Berechnung des Signifikanzfaktors für jeden Satz
4. Auswahl der Sätze mit den größten Signifikanzfaktoren

3.1.3.3.1. Berechnung der Worthäufigkeiten

Da im Extraktionsansatz nach Luhn das einzige Satzauswahlkriterium das Vorhandensein von häufig vorkommenden Worten ist, muss zu erst für jedes Wort die Häufigkeit, mit der es im Text auftritt, berechnet werden. Dies wird in der Methode bestimmeWortHaeufigkeiten() erledigt, welche aus dem übergebenen Text einen Vektor von Wortobjekten erzeugt.
Dazu wird jeder Satz des Textes in die einzelnen Worte zerlegt. Für jedes Wort wird überprüft, ob es im Wortvektor schon enthalten ist. Ist das der Fall wird die Häufigkeit des Wortes, die im Wortobjekt gespeichert ist, um eins erhöht. Ist das Wort noch nicht Bestandteil dieses Vektors, wird ein neues Wortobjekt mit diesem Wort und der Häufigkeit 1 erzeugt und an den Wortvektor angehängt. Bei der Suche nach dem Wort im Wortvektor werden nur dann zwei Worte als identisch angesehen, wenn sie – bis auf die Großschreibung – exakt gleich sind. Sinnvoller wäre die Nutzung eines Tools, welches unterschiedliche Formen eines Wortes vorher auf eine Stammform reduziert. Auch von großem Nutzen wäre ein Tool zur Erkennung von Synonymen.

3.1.3.3.2. Bestimmung signifikanter Worte

Anhand der im vorigen Schritt gewonnenen Informationen über die Wortfrequenzen müssen nun die signifikanten Worte bestimmt werden. Da Worte mit zu häufigem Auftreten als „zu allgemein“ gelten und Worte mit sehr seltenem Auftreten als „unwichtig“, müssen diese aus dem Wortvektor entfernt werden. In der Methode entferneAllgemeineWorte() werden aus dem übergebenen Wortvektor sowohl die zu allgemeinen Worte als auch die zu selten vorkommenden Worte entfernt. Für die Entfernung der zu allgemeinen Worte gibt es zwei unterschiedliche Verfahren:

- Entfernung basierend auf einer Wortliste
- Entfernung basierend auf einem Häufigkeitsgrenzwert

Welches Verfahren verwendet werden soll, wird mit dem zweiten Eingabeparameter bestimmt.

Wortlisten-Verfahren:
Aus der Datei AllgemeineWorte.txt wird eine Wortliste initialisiert, die all die Worte enthält, die als „zu allgemein“ gelten. Diese Textdatei haben wir auf Grundlagen von Literaturstudien und Experimenten erstellt. Es sind zum Beispiel enthalten:

- Artikel
- Präpositionen
- Verschiedene Wortformen der Hilfsverben wie „sein“, „haben“, „sollen“, „werden“ usw.
- Konjunktionen
- Frageworte

Kommt ein Wort des Wortvektors in dieser Wortliste vor, so wird es aus dem Wortvektor gelöscht, da diese allgemeinen Worte nicht signifikant sind.

Häufigkeitsgrenzwert-Verfahren:
Diese Vorgehensweise basiert auf der Annahme, dass Worte, die zu häufig in einem Text vorkommen, für diesen Text nicht sehr aussagekräftig sind. Daher werden aus dem Wortvektor all die Worte gelöscht, die häufiger als 20-mal in einem Text vorkommen. Als Verbesserung wäre hier ein Wert denkbar, der von der Textgröße des Originals abhängt.

Zur Entfernung zu selten vorkommender Worte wird ebenfalls ein Grenzwert eingesetzt. Liegt die Häufigkeit des Auftretens eines Wortes unter 5, wird dieses Wort aus dem Wortvektor gelöscht.

Der so verringerte Wortvektor, welcher nun nur noch signifikante Worte enthält, wird zurückgegeben. Wenn der Zusammenfassungsauftrag bestimmte Suchworte enthält, werden diese an den Wortvektor angehangen, da es sein kann, dass diese bei einem Entfernungsverfahren aus dem Vektor gelöscht wurden.

3.1.3.3.3. Bestimmung des Signifikanzfaktors

Für die Berechnung des Signifikanzfaktors erstellten wir eine neue Klasse SatzZerlegung. In den Objekten dieser Klasse wird ein Satz in seine einzelnen Wort zerlegt gespeichert und anhand dieser Zerlegung und dem Wortvektor mit den signifikanten Worten der Signifikanzwert des Satzes ermittelt.

SatzZerlegung
-anzWorte:int - ID:int -signifikanzFaktor:double -worte:Vector
+ berechneSignifikanzFaktor(signifikanteWorte:Vector):double + getSatzID():int + getSignifikanzFakor():double + getWortAnStelle(index:int):String + istWortInVektorEnthalten(gesuchtesWort:String,signifikanteWorte:Vektor):boolean + signifikanzFunktion(anzSignWorte:int,anzGesWorte:int):double

Abbildung 17: Klassendiagramm der Klasse SatzZerlegung

Die berechneSignifikanzFaktor()-Methode durchläuft den zerlegten Satz Wort für Wort. Wird dabei ein signifikantes Wort entdeckt, wird in einem Integer-Feld an der Stelle, an der das Wort im Satz steht, eine 1 eingetragen, sonst eine 0. Dieses Hilfsfeld dient der Ermittlung von signifikanten Wortgruppen, wofür die Abstände zwischen den einzelnen signifikanten Worten benötigt werden.

[0 0 1 0 0 0 0 0 1 0 0 1 1 0 0 0 1 0 0 1 0]

Abbildung 18: Interne Repräsentation eines Satzes mit signifikanten Worten

Bei dem anschließenden Durchlauf durch dieses Hilfsfeld wird bei der ersten 1 (entspricht dem ersten signifikanten Wort) der potenzielle Anfang einer signifikanten Wortgruppe vermerkt.

Pot. Anfang einer Wortgruppe

[0 0 **1** 0 0 0 0 0 1 0 0 1 1 0 0 0 1 0 0 1 0]

Abbildung 19: Entdeckung des ersten signifikanten Wortes im Satz

Findet man innerhalb der nächsten 5 Worte nicht ein zweites signifikantes Wort, wird diese potenzielle Wortgruppe verworfen. Wird jedoch eine weitere 1 entdeckt, wird an dieser Stelle das potentielle Ende der Wortgruppe vorgemerkt.

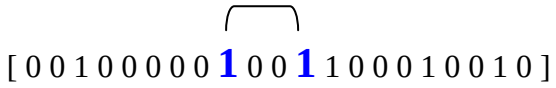


Abbildung 20: Entdeckung der ersten signifikanten Wortgruppe

Findet man unter den nächsten 5 Worten wiederum eine 1, wird die Wortgruppe bis dahin verlängert. Dies wird solang fortgesetzt bis der Satz zu Ende ist oder im Abstand von 5 Worten kein weiteres signifikantes Wort gefunden wurde.

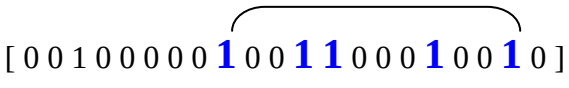


Abbildung 21: Vollständige signifikante Wortgruppe wurde ermittelt

Wurde erkannt, dass eine signifikante Wortgruppe vollständig ist, entweder da innerhalb der folgenden 5 Worte kein weiteres signifikantes Wort aufgetaucht ist oder da das Ende des Satzes erreicht ist, wird für diese Wortgruppe der Signifikanzfaktor berechnet und mit dem alten Signifikanzfaktor der SatzZerlegung verglichen. Ist der neu berechnete Signifikanzfaktor größer, wird dieser als Signifikanzfaktor für dieses SatzZerlegungs-Objekt gespeichert.
Der Signifikanzfaktor berechnet sich nach folgender Formel:

$$\text{Signifikanzfaktor} = \frac{(\text{Anzahl signifikanter Worte in der Wortgruppe})^2}{\text{Anzahl Worte in der Wortgruppe}}$$

Anschließend wird das Hilfsfeld auf der Suche nach weiteren signifikanten Wortgruppen durchlaufen. Für diese Wortgruppen werden die oben erwähnten Schritte (Finden des potenziellen Wortgruppenanfangs, Finden des Endes der Wortgruppe, Verlängern der

Wortgruppen, Berechnen des Signifikanzfaktors der Wortgruppe) durchgeführt, bis das Ende des Feldes – also des Satzes – erreicht wird. So erhält die SatzZerlegung dieses Satzes den größten Signifikanzfaktor aller enthaltenen, signifikanten Wortgruppen.

3.1.3.3.4. Bestimmung der signifikanten Sätze

Nachdem für jeden Satz der Signifikanzfaktor berechnet wurde, werden die Sätze nach absteigendem Signifikanzfaktor geordnet. Nun werden die höchstsignifikanten Sätze ausgewählt, so dass die Anzahl von Sätzen herauskommt, die durch Originaltextgröße und Kompressionsrate bzw. der eingegeben Zusammenfassungssatzanzahl vorgegeben ist. Diese Sätze werden nach aufsteigender SatzID geordnet, damit die Sätze in derselben Reihenfolge wie im Originaltext vorliegen.

3.1.3.4. Extraktionsalgorithmus nach Edmundson

Die Auswahl der relevanten Sätze erfolgt bei diesem Algorithmus nach Vorbild des Edmundsonalgorithmus von 1969. Dieser Algorithmus verwendet 4 verschiedene Methoden zur Ermittlung der Satzgewichte:

- Location-Methode
- Title-Methode
- Key-Methode
- Cue-Methode

Daher erhält die edmundsonAlgorithmus()-Methode neben dem Text auch ein Objekt der Klasse ParameterEdmundson als Eingabewert, welches die Gewichte für die Werte der vier Methoden bei der Berechnung des Satzgewichtes enthält. Der Rückgabewert ist auch hier ein Integer-Feld, welches die SatzIDs der relevanten Sätze enthält.

3.1.3.4.1. Berechnung des Locationgewichtes

Bei der Berechnung des Locationgewichtes für einen Satz werden die Position des Satzes und die Überschrift, unter der er steht berücksichtigt. Die berechneLocationGewicht()-Methode erhält dafür die SatzID des betroffenen Satzes und den gesamten Text als Text-Objekt.
Zuerst wird das Headingdictionary aus einer Datei geladen. In diesem Dictionary sind die Worte mit Gewichten enthalten, die oft in Überschriften vorkommen und dadurch eine gewisse Wichtigkeit des folgenden Abschnittes anzeigen. Das zu jedem Wort gespeicherte Gewicht gibt einen Hinweis auf den Grad an Wichtigkeit.

Zusammenfassung	0.5	Vorteil	0.3
Einleitung	0.5	Nachteil	0.3
Einführung	0.5	Abstract	0.4
Literatur	0.2	Ergebnis	0.2
Folgerung	0.3	Diskussion	0.3
Schlussfolgerung	0.3		

Abbildung 22: Auszug aus dem Headingdictionary

Nachdem die Überschrift des Abschnittes ermittelt wurde, in welchem der Satz steht, wird nun überprüft, welches Wort des Headingdictionarys in der Überschrift enthalten ist. Wird ein Headingdictionary-Wort in der Überschrift entdeckt, addiert man das Gewicht dieses Wortes zum bisherigen Locationgewicht. Nachdem so der Teil des Locationgewichtes, der sich aus der Überschrift ableitet, bestimmt wurde, wird überprüft, ob der betroffene Satz eventuell im ersten oder letzten Abschnitt steht und ob er eventuell der erste oder letzte Satz eines Abschnittes ist. Für jeden dieser vier Fälle wurde ein bestimmtes Gewicht festgelegt, welches zum bisherigen Locationgewicht addiert wird, wenn dieser Fall eintritt. Dabei kann ein Satz auch gleichzeitig mehrere wichtige Positionen im Text oder Abschnitt erfüllen. Wenn ein Satz zum Beispiel der erste Satz des letzten Abschnitts ist, erhält er das Gewicht für den letzten Abschnitt und das Gewicht für den ersten Satz.

3.1.3.4.2. Berechnung des Titlegewichtes

Auch bei der Ermittlung des Titlegewichtes spielen Überschriften eine Rolle. Allerdings sind hier die tatsächlichen Überschriften des Textes und der Inhalt des betroffenen Satzes Gegenstand der Untersuchung. Bevor das Titlegewicht eines Satzes bestimmt werden kann, muss das Titleglossar geladen werden. In diesem Glossar sind alle Worte, die im Titel des Textes oder in Überschriften und Zwischenüberschriften des Textes vorkommen, mit dazugehörigen Gewichten gespeichert. Allgemeine Worte wie Artikel, Konjunktion, Hilfsverben oder ähnliches wurden aber nicht im Glossar gespeichert. Nun wird dieses Glossar Wort für Wort durchgegangen und überprüft, ob das gerade betrachtete Wort im Satz, für den gerade das Titlegewicht berechnet werden soll, enthalten ist. Wenn dies der Fall ist, wird das Gewicht dieses Wortes, welches im Glossar gespeichert ist, zum bisherigen Titlegewicht des Satzes addiert. Wurden so alle Worte des Titleglossars abgearbeitet, steht das Titlegewicht des Satzes fest.

3.1.3.4.3. Berechnung des Keygewichtes

Die Keygewichtberechnung basiert auf der Annahme, dass häufig erwähnte Worte wichtig für den Text sind. Für die Ermittlung des Keygewichtes erhält die berechneKeyGewicht()-Methode die SatzID des Satzes, für den das Keygewicht bestimmt werden soll, den Text und das Keyglossar. In diesem Glossar sind die Worte des Textes und deren Häufigkeiten enthalten, die nicht als positive bzw. negative Hinweisworte gelten, die nicht als allgemeine Worte gelten und deren Häufigkeit über einem bestimmten Grenzwert liegt.

Zur Bestimmung des Keygewichtes wird für jedes Wort dieses Glossars überprüft, ob es im Satz vorkommt. Ist das Wort im Satz enthalten, wird die Häufigkeit des Wortes als Gewicht zum bisherigen Keygewicht addiert.

3.1.3.4.3.1. Erstellen des Keyglossars

Im Gegensatz zu den anderen Edmundson-Methoden wird hier das benötigte Glossar nicht in der Gewichtsberechnungsmethode erstellt, sondern in der aufrufenden Methode und als Parameter übergeben. Dies liegt daran, dass die Erstellung des Keyglossars aufwendiger ist. In der ladeKeyGlossar()-Methode werden zuerst aus den Dateien zwei Dictionarys erstellt. Das eine enthält alle so genannten Bonusworte, das sind die positiv bzw. negativ hinweisenden Worte. Das andere enthält die allgemeinen Worte wie Konjunktionen, Artikel, Pronomen und so weiter. Nachdem für alle Worte des Textes die Anzahl ihres Vorkommens ermittelt wurde und zusammen mit dem Wort in dem Keykandidaten-Vektor hinterlegt wurde, werden die Worte wieder aus dem Vektor entfernt, die in einem der beiden Dictionarys enthalten sind. Dann wird ein Grenzwert für die Gesamthäufigkeit aus dem Gesamtwortanzahl und einer Prozentangabe ermittelt. In ein Feld Keyglossar werden nun solange die nach absteigender Häufigkeit geordneten Worte übernommen, bis die Summe der Häufigkeiten der übernommenen Worte über dem Grenzwert liegt.

3.1.3.4.4. Berechnung des Cuegewichtes

Die Grundlage für das Cuegewicht bildet die Annahme, dass bestimmte Worte oder Phrasen die Relevanz oder Nicht-Relevanz eines Satzes anzeigen. Deshalb wird als erstes in der berechneCueGewicht()-Methode das Cuedictionary geladen, welches diese so genannten Cueworte – Bonus- und Stigmaworte – enthält. Die Cueworte sind darin mit ihren Gewichten gespeichert, welche anzeigen, ob ein Wort auf die Relevanz oder auf die Nicht-Relevanz eines Satzes hinweist:

- Gewicht > 0: positiv relevant (Bonuswort)

- Gewicht < 0: negativ relevant (Stigmawort)

Nun wird getestet, welche dieser Bonusworte im Satz enthalten sind. Die Gewichte der enthaltenen Worte werden zum Cuegewicht des Satzes aufsummiert.

3.1.3.4.5. Berechnung des endgültigen Satzgewichtes

Für das Berechnen des endgültigen Satzgewichtes werden die übergebenen Methodengewichte benötigt.

$$\begin{aligned} \text{Satzgewicht}(s) = \quad & \text{Gewicht}_{\text{Cue}} * \text{CueGewicht}(s) + \\ & \text{Gewicht}_{\text{Key}} * \text{KeyGewicht}(s) + \\ & \text{Gewicht}_{\text{Title}} * \text{TitleGewicht}(s) + \\ & \text{Gewicht}_{\text{Location}} * \text{LocationGewicht}(s) \end{aligned}$$

Abbildung 23: Berechnungsformel für das Satzgewicht des Satzes s nach dem Edmundsonalgorithmus

Durch die Möglichkeit der unterschiedlichen Gewichtungen können die Satzgewichte zum Beispiel auch nur durch eine Methode berechnet werden oder durch die Kombination von zwei oder drei Methoden.

3.1.3.4.6. Auswahl der relevanten Sätze

Wie bei den vorangegangenen Extraktionsalgorithmen werden die Sätze nach ihren Satzgewichten absteigend sortiert und die Anzahl an Sätzen mit den höchsten Gewichten ausgewählt, die durch die Kompressionsrate und die Textgröße vorgeben wird.

3.1.3.4.7. Vereinfachungen gegenüber dem Original-Edmundson-Algorithmus

Auf Grund eines fehlenden Tools zur Stammformenreduktion sind verschiedene Formen des gleichen Wortes im Keyglossar enthalten.

Die Bonus- und Stigmaworte sowie deren Gewichte und die allgemeinen Worte, auch Nullworte genannt, erhielt Edmundson ursprünglich in dem er einen großen Textkorpus auswertete. Dafür berechnete er in 100 Dokumenten für jedes Wort folgende Werte:

- Frequenz (Anzahl an Vorkommen des Wortes im Korpus)
- Dispersion (Anzahl der Dokumente, in denen das Wort vorkommt)
- Selektionsverhältnis (Anzahl an Vorkommen des Wortes in manuell für eine Zusammenfassung extrahierten Sätzen)

Aus diesen Werten leitete er die folgenden Wortkategorien ab:

- Nullwort: wenn die Dispersion über einem bestimmten Wert liegt und das Selektionsverhältnis zwischen zwei gewählten Grenzwerten liegt
- Bonuswort: wenn das Selektionsverhältnis über einem festgelegten Wert ist
- Stigmawort: wenn das Selektionsverhältnis unter einem festgelegten Wert ist
- Rest: wenn die Dispersion unter einem bestimmten Wert liegt und das Selektionsverhältnis zwischen zwei gewählten Grenzwerten liegt

Da wir nicht auf einen entsprechenden Korpus an deutschsprachigen Texten und dazugehörigen Zusammenfassungen zurückgreifen konnten, untersuchten wir einige wenige deutsche Texte und entschieden manuell, welche Worte oder Wortarten zu allgemein sind und welche Worte oder Wortgruppen auf wichtige bzw. unwichtige Sätze hinweisen könnten. Ähnlich verfuhren wir bei der Herausarbeitung von Überschriftsworten, welche auf die Relevanz des nachfolgenden Abschnittes verweisen. Die Gewichte der Cueworte und der Überschriftsworte wurden dann intuitiv bestimmt. So waren wir der Meinung, dass Überschriften, die Worte wie „Zusammenfassung“ oder „Einleitung“ enthalten, wichtig sind und daher höhere Gewichte erhalten müssten als Überschriften die Worte wie „Nachteil“ oder „Vorteil“ enthalten, da die letzteren nur auf einen Abschnitt mit einseitigen, die ersten aber auf Paragraphen mit umfassende Informationen verweisen werden.

3.1.3.5. Extraktionsalgorithmus nach Euler

Timm Euler nutzt in seinem Extraktionsalgorithmus Stichwortlisten zu bestimmten Domänen, um die wichtigsten Sätze zu extrahieren. Nach diesem Vorbild implementierten wir den vierten Extraktionsalgorithmus. Die Methode eulerAlgorithmus() erhält dafür den Text als Eingabe und liefert die relevanten Sätze als SatzIDs in einem Integer-Feld zurück.

3.1.3.5.1. Erstellen der Stichwortliste

Euler ließ die Stichwortlisten in relativ aufwendigen Verfahren generieren. Er hatte einen großen Textkorpus zur Verfügung und markierte in einigen Texten zu seinem Thema relevante Sätze. Für alle Worte dieser Sätze berechnete er mit verschiedenen Methoden Gewichte und ließ nach diesen Gewichten geordnete Listen erstellen.

Wir vereinfachten das Verfahren in dem wir in 2 Texten manuell die wichtigsten Sätze markierten, alle Worte dieser Sätze speicherten, diese Worte auf ihre Stammformen reduzierten und die Worte anschließend folgendermaßen gewichteten:

$$\text{Gewicht}(w) = \frac{\text{Anzahl der Vorkommen von w in positiven Sätzen}}{\text{Anzahl der Vorkommen von w in negativen Sätzen}}$$

Abbildung 24: Berechnungsvorschrift für das Wortgewicht des Wortes w

Wir entschieden uns für diese Gewichtung, da wir nicht auf einen großen Textkorpus zurückgreifen konnten und daher nicht die Häufigkeit des Wortes in nicht zum Thema passenden, so genannten negativen Texten ermitteln konnten.

Bei der Stammformreduktion speicherten wir für jedes Wort den Wortstamm in den verschiedenen Zeitformen bzw. im Singular und Plural. (vgl. Abbildung 25 und Abbildung 26) Bei einigen Worten speicherten wir auch die englische Schreibweise, wenn diese oft im Deutschen verwendet wird, wie zum Beispiel Kode-Code oder Kompression-Compression.

Wort	Anz. in pos. Sätzen	Anz. im Rest
Abbilden, bildet ab, abgebildet	2	2
adaptiven	1	7
Ähnliche	1	1
Anwenden	1	1
Anzahl	1	2
anzunehmen, Annahme, angenommen, annehmen	3	2
arithmetische	3	5
aufteilten	1	2
Auftretens, auftretenden, Auftritt, tritt auf	5	8
Ausgegangen	1	1
auszuwählen	1	1
begann, Beginn, beginnend	2	9
Bekannter, bekannt	2	2

Abbildung 25: Auszug aus der Liste der markierten Worte und deren Häufigkeiten in positiven und negativen Sätzen

Wort	Anz. In pos. Sätzen	Anz. Im Rest0	Gewicht
abbild, bildet, abgebildet	2	2	1,00000
adaptiv	1	7	0,14286
ähnlich	1	1	1,00000
anwend, angewendet	1	1	1,00000
Anzahl	1	2	0,50000
anzunehmen, Annahme, angenommen, annehmen	3	2	1,50000
arithmetisch	3	5	0,60000
aufteil, aufgeteil	1	2	0,50000
auftret, aufgetret, Auftritt, tritt	5	8	0,62500
ausgegangen, ausgehen	1	1	1,00000
auszuwählen, auswähl, auswahl	1	1	1,00000
begann, beginn, begonnen	2	9	0,22222
bekannt	2	2	1,00000

Abbildung 26: Auszug aus der Liste der stammformreduzierten Worte mit Wortgewichten

Wir entschieden uns für den Bereich „Kompression und Kodierung“ als Domäne für unsere Stichwortliste. Die untersuchten Texte behandelten die Themen „Huffman-Kodierung“ und „JPEG-Kompression“.

Die auf diese Art und Weise ermittelten Stichworte und deren Gewichte wurden in einer Textdatei durch Tabstops getrennt gespeichert. Um eventuell andere Stichwortlisten aus anderen Domänen nutzen zu können, kann in der GUI der Pfad zur Stichwortdatei geändert werden, allerdings müssen die neuen Stichwortdateien die gleiche Formatierung aufweisen.

3.1.3.5.2. Berechnen der Satzgewichte

Um das Gewicht jedes einzelnen Satzes zu ermitteln, wird als erstes die Stichwortliste eingelesen. Dazu wird jeweils eine Wortstammgruppe mit ihrem Gewicht in einem Objekt der Klasse Stichwort gespeichert. Diese Stichwortobjekte werden dann in einem Vektor gespeichert.

Jeder Satz wird in seine einzelnen Worte zerlegt und es wird für jedes Wort überprüft, ob dessen Wortstamm im Stichwortvektor enthalten ist. Wenn dies der Fall ist, wird das Gewicht des übereinstimmenden Wortstamms zum bisherigen Satzgewicht addiert. Wurden alle Worte eines Satzes betrachtet, dividiert man das Satzgewicht durch die Anzahl der im Satz enthaltenen Worte, um so lange Sätze nicht zu bevorzugen.

$$\text{Satzgewicht}(s) = \frac{\text{Summe der Gewichte der enthaltenen Stichworte}}{\text{Anzahl der Worte im Satz s}}$$

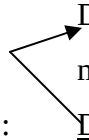
Abbildung 27: Berechnungsformel für das Satzgewicht des Satzes s

3.1.3.5.3. Bestimmung der zu selektierenden Sätze

Die Sätze für die Zusammenfassung wurden wie bei den vorherigen Algorithmen ausgewählt.

3.1.3.6. Extraktionsalgorithmus nach unseren Vorstellungen

Auf Grundlage der Analyse der vorher erwähnten Auswahlalgorithmen und durch eigene Überlegungen haben wir die Selektionskriterien für unseren Algorithmus bestimmt.
Sinnvoll erscheint uns die Benutzung der so genannten Hinweisphrasen, welche auf wichtige aber auch auf unwichtige Sätze hinweisen können. Auch die Berücksichtigung von bestimmten Schlüsselworten in Überschriften bringt unserer Meinung nach wertvolle Hinweise in Bezug auf die Bedeutung des dazugehörigen Abschnitts. Um nicht nur allgemeine sondern auch auf Nutzerinteressen ausgerichtete Zusammenfassungen erstellen zu können, beachteten wir bei der Satzselektion auch die im Satz oder der Überschrift enthaltenen Suchworte, welche vom Nutzer eingegeben wurden, um seine Interessen zu verdeutlichen. Um die Anzahl von Verweisen im Text, die auf nicht-selektierte Sätze verweisen, zu verringern, wurden auch solche Sätze als wertvoll deklariert, auf die ein wichtiger Satz verweist. Folgendes Beispiel soll diese Selektionskriterium verdeutlichen:



Satz x: Das Huffman-Verfahren liefert nachweisbar präfixfreien Kode und minimiert die kodierte Nachricht.

Satz x+1: Dieses Verfahren hat viele Nachteile, welche durch das adaptive Verfahren, bei dem sich der Kode ständig verändert, beseitigt werden können.

Abbildung 28: Unter der Annahme, dass der Satz x+1 als wichtig gilt, erhält auch der Satz x ein höheres Gewicht, das sich der Satz x+1 auf den vorhergehenden Satz bezieht.

Da Sätze, die Verweise auf andere Sätze enthalten, auch selbst wertvoll für eine Zusammenfassung sind, erhalten sowohl der verweisende Satz als auch der Satz, auf den verwiesen wird, für diese Beziehung einen bestimmten Gewichtswert.
Weil wir mit unserem System auch Internet-Seiten zusammenfassen können, wollten wir auch die Besonderheiten dieses Datentyps für unsere Zwecke nutzen. So wurden Sätze die Links

enthalten, welche die eingegebenen Suchworte oder andere Schlüsselworte (z.B. „Zusammenfassung“,…) beinhalten, als wichtig für die Zusammenfassung erachtet.
Außerdem beachten wir die Position im Text und im Abschnitt sowie die Länge und die Art des Satzes.

3.1.3.6.1. Bestimmung der Satzgewichte

Zur Ermittlung der zu selektierenden Sätze entschieden wir uns für ein Satzgewichtverfahren, bei dem das Gewicht eines Satzes die Summe verschieden wichtiger Satz- und Abschnittsmerkmale ist.

```
Satzgewicht(s) =    Gewicht * SchlüsselwortWert(s) +
                    Gewicht * PositionImAbschnittWert(s) +
                    Gewicht * PositionImTextWert(s) +
                    Gewicht * SuchwortWert(s) +
                    Gewicht * SuchwortInÜberschriftWert(s) +
                    Gewicht * SatzrelationsWert(s) +
                    Gewicht * EigennamenWert(s) +
                    Gewicht * LinkWert(s) +
                    Gewicht * BildWert(s) +
                    Gewicht * SatzlängenWert(s) +
                    Gewicht * WichtigeKonjunktionenWert(s) +
                    Gewicht * WichtigeÜberschriftenWert(s) +
                    Gewicht * FragenWert(s)
```

Abbildung 29: Berechnung des Satzgewichtes nach unserem Algorithmus

3.1.3.6.1.1. Bestimmung des SchlüsselwortWertes

Die Basis für den SchlüsselwortWert ist die Annahme, von der auch Edmundson ausging, dass es Worte gibt, die die Wichtigkeit des Satzes anzeigen. Als Schlüsselworte bezeichnen wir bei unserem Extraktionsalgorithmus die Worte, die sowohl auf wichtige als auch auf unwichtige Sätze hinweisen (vgl. Cueworte bei Edmundson - Kapitel 2.3.1.2.). Diese Worte sind zusammen mit ihren Gewichten in einer Textdatei gespeichert, so dass die Liste der Schlüsselworte auch verändert oder ergänzt werden kann. Schlüsselworte, die auf wichtige Sätze hinweisen, haben ein positives Gewicht. Die Schlüsselworte, die auf unwichtige Sätze deuten, erhielten negative Gewichte. Der SchlüsselwortWert eines Satzes berechnet sich daher als Summe der Gewichte der Schlüsselworte, die in diesem Satz enthalten sind.

3.1.3.6.1.2. Bestimmung des PositionImAbschnittWertes

Grundlage für diesen und den folgenden Wert, ist unsere Annahme, dass Sätze am Anfang des Textes und am Anfang eines Abschnittes potentiell wichtig sind. Dies beruht auf der Vermutung, dass die meisten Texte, die zusammengefasst werden sollen, Argumentationen sind und somit dem einfachen oder dem erweiterten Argumentationsschema entsprechen. Bei diesen Schemata folgt auf die am Anfang des Textes stehende These in jedem Abschnitt ein Argument, welches mit Beispielen belegt wird. Daher gehen wir davon aus, dass in den ersten fünf Sätzen jedes Abschnittes ein Argument dargelegt wird. Da die Argumente als die Hauptaussagen eines Textes betrachtet werden können, erhalten die ersten fünf Sätze eines Abschnittes einen positiven Gewichtswert.

$$\text{PositionImAbschnittWert}(s) = \begin{cases} 0.5 / \text{PositionImAbschnitt}(s); & \text{PositionImAbschnitt} <= 5 \\ 0; & \text{sonst} \end{cases}$$

Abbildung 30: Berechnungsformel zur Bestimmung des PositionImAbschnittWertes für Satz s

3.1.3.6.1.3. Bestimmung des PositionImTextWertes

Wie im vorangegangenen Abschnitt erwähnt, dient dieser Wert dazu, Sätze am Anfang des Textes als extrahierwert zu kennzeichnen. Auch hier ist die Grundlage für diese Theorie das Argumentationsschema. Die ersten fünf Abschnitte sind wichtig, da nach diesem Schema die These des Textes also dessen Kernbehauptung in diesen Abschnitten zu finden ist. Des Weiteren ist es möglich, dass bei der Argumentation mit den stärksten Argumenten begonnen wird. Diese schließen sich dann in den Abschnitten nach der These an und stehen somit auch am Anfang des Textes.

Allerdings sind auch die letzten fünf Abschnitte oftmals von großer Bedeutung, da es auch Argumentationsfolgen gibt, bei denen mit den schwächsten Argumenten begonnen wird und die stärksten Argumente somit erst am Schluss des Textes zu finden sind. Außerdem dient meist der letzte Abschnitt dazu, den gesamten Text kurz zusammenzufassen. In dieser Zusammenfassung sind die wichtigsten Punkte des Textes in knapper Form noch einmal erfasst, so dass diese Sätze sehr viele wichtige Informationen enthalten und daher sinnvoll für eine Zusammenfassung sind. Aus diesen Gründen werden sowohl die ersten fünf als auch die letzten fünf Abschnitte eines Textes mit positiven Gewichten versehen.

$$PositionImTextWert(s)=\begin{cases} 1 / PositionImText(s); & PositionImText(s)<=5 \\ 1/(MaxPos + 1 - PositionImText(s)); & PositionImText(s)>MaxPos - 5 \\ 0; & sonst \end{cases}$$

Abbildung 31: Berechnungsformel zur Bestimmung des PositionImTextWertes für Satz s

3.1.3.6.1.4. Bestimmung des SuchwortWertes und des SuchwortInÜberschriftWertes

Durch die Berücksichtigung von Suchworten in den Sätzen und Überschriften, welche der Nutzer über die GUI eingeben kann, ist es mit unserem Extraktionsalgorithmus möglich, Zusammenfassungen nach Nutzerinteressen zu erstellen. Für jedes gefundene Suchwort in einem Satz oder in der dazugehörigen Abschnittsüberschrift erhält dieser Satz eine Satzgewichtserhöhung von 1. So berechnet sich der SuchwortWert eines Satzes als Summe der enthaltenen Suchworte. Der SuchwortInÜberschriftWert, welcher für alle Sätze eines Abschnittes gleich ist, ist die Summe der in der Überschrift enthaltenen Suchworte.

3.1.3.6.1.5. Bestimmung des SatzrelationsWertes

Der Ausgangspunkt für den SatzrelationsWert ist unsere Vermutung, dass Sätze, die in einer Beziehung zueinander stehen mit großer Wahrscheinlichkeit wichtig sind. Allerdings sind wir der Meinung, dass zwei Sätze, die in einer Relation stehen, nur dann wirklich wichtig sind, wenn der erste Satz, auf den sich der zweite bezieht, schon ein überdurchschnittliches Satzgewicht aufweist.

Daher kann der SatzrelationsWert erst nach allen anderen Werten berechnet werden. Anschließend wird das durchschnittliche Satzgewicht berechnet und für alle Sätze folgendes überprüft:

- Liegt das Satzgewicht des aktuell betrachteten Satzes über dem durchschnittlichen Satzgewicht?
- Beginnt der folgende Satz mit einem Wort, das auf eine Beziehung zwischen den Sätzen hinweist (daher, dadurch, deshalb, dabei, dieser, dieses, …)?
- Sind der aktuell betrachtete Satz und der nachfolgende Satz aus demselben Abschnitt?

Wenn diese drei Bedingungen erfüllt sind, beträgt der SatzrelationsWert für diese Satzbeziehung für beide Sätze 1. Wenn der zweite Satz einer Satzrelation mit dem nächsten Satz auch wieder in einer Beziehung steht, dann erhält er für diese Satzbeziehung abermals einen Wert von 1. Daher berechnet sich der endgültige SatzrelationsWert für einen Satz s folgendermaßen:

$$SatzrelationsWert(s) = RelationsWert(s-1,s) + RelationsWert(s, s+1)$$

$$RelationsWert(x,x+1) = \begin{cases} 1; & \text{Bed. 1 und Bed. 2 und Bed. 3 erfüllt} \\ 0; & \text{sonst} \end{cases}$$

Abbildung 32: Berechnungsformel für den SatzrelationsWertes des Satzes mit der SatzID s

3.1.3.6.1.6. Bestimmung des EigennamenWertes

Wir halten Sätze die Eigennamen enthalten für extrahierwert, da in diesen Sätzen Namen von Personen, Unternehmen, Forschungsgruppen und ähnlichem erwähnt werden und diese meist mit dem Thema des Textes stark verbunden sind.

Eigennamen definieren wir dabei als „Substantiv oder substantivische Wortgruppe (onymische Benennung) mit der Funktion der Identifizierung des einzelnen Objekts und dessen Differenzierung von anderen gleichartigen Objekten (Leipzig, Saale, Altes Rathaus, Juliane)“ [Kluckhohn 2004].

Um den EigennamenWert eines Satzes bestimmen zu können, mussten wir uns überlegen, wie wir Eigennamen erkennen können. In englischen Texten ist dies kein Problem, da dort fast alle großgeschriebenen Worte Eigennamen sind. Da im Deutschen allerdings alle Substantive großgeschriebenen werden, war es für uns nicht so einfach zwischen Substantiven und Namen zu unterscheiden. Wir untersuchten dazu einige Eigennamen in Texten und erzeugten daraufhin in Java ein Pattern, welches als Schablone für Eigennamen dient und diese erkennt. So erkennen wir alle die Wortgruppen als Eigennamen, die entweder mindestens zwei großgeschriebene Worte oder mindestens eine Initiale und ein großgeschriebenes Wort direkt hintereinander enthalten. Für jede solche Wortgruppe wird das Gewicht eines Satzes um eins erhöht.

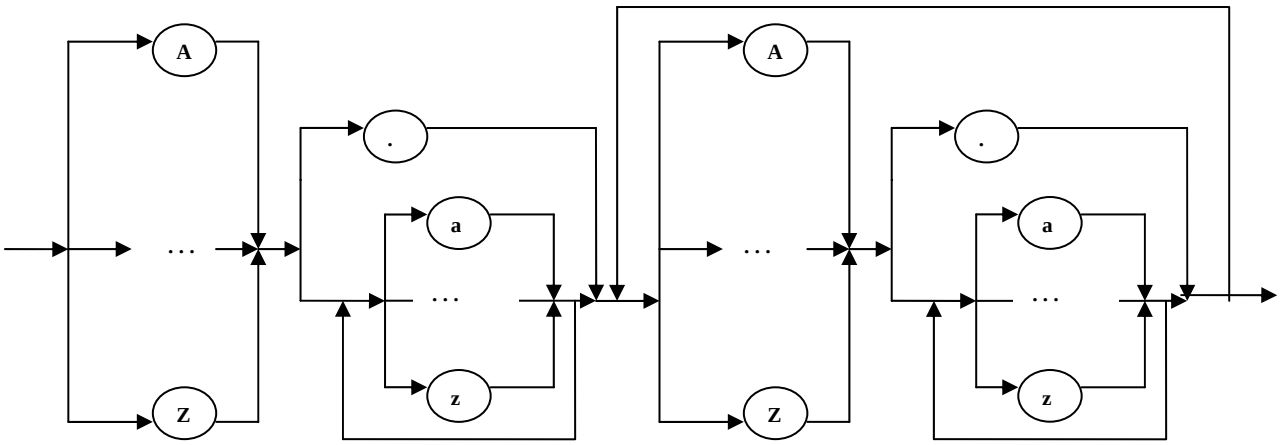


Abbildung 33: Eigennamenregel als Syntaxdiagramm

Allerdings erkennt man mit dieser einfachen Methode nicht nur Eigennamen. Steht zum Beispiel als zweites Wort im Satz ein beliebiges Substantiv, so wird dieses zusammen mit dem ersten Wort des Satzes, welches ja auch großgeschrieben wird, als Eigenname angesehen. Aber auch zwei unterschiedliche Substantive, die nacheinander in einem Satz auftauchen, werden so fälschlicher Weise als Name behandelt.

Die Dekodierung einer kodierten Nachricht verläuft sehr ähnlich:
Verfolgt man aber Shannons Idee konsequent weiter …
… muß man sich zuerst überlegen, in welcher Form Redundanz auftreten kann.

Abbildung 34: Beispiele für falsch erkannte Eigennamen

Da sehr häufig Fehler bei der Eigennamenerkennung wie im ersten Beispielsatz auftraten, änderten wir unsere Methode dahingehend, dass nun überprüft wird, ob das erste Wort des potentiellen Eigennamens ein Artikel ist. Wenn dies der Fall ist, verwerfen wir diesen Eigennamenkandidaten.
Was mit diesem Verfahren allerdings nicht erkannt werden kann, sind Eigennamen, die nur aus einem Wort bestehen wie die bloße Benutzung des Nachnamens einer Person. Dies zu Realisieren ist nur sehr schwer möglich, denn selbst wenn man für jedes großgeschriebene Wort prüft, ob es ein „normales“ Substantiv ist oder ein Eigenname, so können Personen die mit Nachnamen beispielsweise Baum oder Berg heißen, nicht herausgefiltert werden.
Natürlich gibt es bessere Systeme zur Eigennamenerkennung wie zum Beispiel Proper Name Facility (PNF) von SPARSER oder BSEE von FACILE / CONCERTO, welche entweder mit Hilfe von Listen oder durch begrenzte Domänen Eigennamen identifizieren. Allerdings würde

die Umsetzung eines solchen Systems nur für die Berechnung des EigennamenWertes den Rahmen sprengen.

3.1.3.6.1.7. Bestimmung des LinkWertes

Da unser Zusammenfassungssystem neben TXT- und RTF-Texten auch HTML-Dokumente zusammenfassen kann, wollten wir die Besonderheiten dieses Formates wie die Verlinkung berücksichtigen. Die Möglichkeit Links zu folgen, eröffnet den Zugang zu weiteren eventuell brauchbaren Informationen. Allerdings mussten wir eine Möglichkeit finden, wie wir erkennen können, welche Links sinnvoll sind und daher mit in die Zusammenfassung übernommen werden sollten.

Da das Ziel eines Verweises, welches zwischen <a> und steht, dem Nutzer einen Hinweis geben soll, was ihn bei Verfolgung dieses Links erwartet, entschieden wir uns, dass dieser Text, von uns Verweiszieltext genannt, einen Anhaltspunkt für die Wichtigkeit des Links bietet. So wird im Verweiszieltext nach den übergebenen Suchworten, aber auch nach bestimmten Schlüsselworten wie „Zusammenfassung“, „Einleitung“ oder „Vorteil“ gesucht. Für jedes gefundene Such- oder Schlüsselwort erhält der Satz eine Gewichtserhöhung von 1.

Da wir aber feststellten, dass sehr oft nichts sagende Verweiszieltexte wie „weiter“, „zurück“ oder „nächste“ verwendet werden, betrachteten wir auch das Verweisziel, welches als Attribut zum HREF-Element angegeben wird. Oftmals haben die HTML-Dateien, auf die verwiesen wird aussagekräftige Namen wie zum Beispiel „/wiki/Moving_Picture_Experts_Group“ oder „http://www.educeth.ch/informatik/vortraege/xmlintro/“, die auf Seiten über MPEG oder XML verweisen. Daher überprüfen wir auch die Verweisziele auf das Vorkommen von Such- und Schlüsselworten.

Um aber noch aussagekräftiger Entscheidungen treffen zu können, entschieden wir uns, da wo es möglich ist, auch den Inhalt der Internetseite, auf die verlinkt wird, zu betrachten. Dies ist allerdings nur möglich, wenn eine vollständige URI (URI = Universal Resource Identifier) als Verweisziel angegeben ist. Bei Referenzierung mit absoluten oder relativen Pfadangaben wäre es notwendig, die URL der HTML-Seite zu wissen. Dies ist aber bei lokal gespeicherten Internetseiten nicht möglich.

Vollständige URIs erkennt unser Algorithmus daran, dass sie mit „http“ oder „ftp“ beginnen. Wurde eine solche Adresse gefunden, wird überprüft, auf welches Dateiformat verlinkt wird. Unser Programm analysiert nur den Inhalt von HTML-Referenzen, so dass nur bei Adressen, die mit „.htm“ oder „.html“ enden, eine weitere Untersuchung des Inhalts der Zielseite des Links stattfindet.

Mit der Klassenmethode holeInhaltAusWeb() der Klasse AllgemeineMethoden wird der Inhalt der Internetseite, auf die der Link verweist, in einem String gespeichert und so an unseren Extraktionsalgorithmus zurückgegeben. Nun wird in diesem String nach den Suchworten des Nutzers gesucht. Ist die Anzahl der gefundenen Suchworte größer als ein vorgegebener Wert, erhält der Satz, der diesen Link enthält, eine Gewichtserhöhung von 5.
So berechnet sich der LinkWert eines Links als Summe aus den folgenden drei Komponenten:

- Summe der enthaltenen unterschiedlichen Suchworte im Link
- Summe der enthaltenen unterschiedlichen Schlüsselworte im Link
- $\begin{cases} 5; \text{ wenn Anzahl gefundener Suchworte auf der Zielseite größer als bestimmter Wert} \\ 0; \text{ sonst} \end{cases}$

Der LinkWert eines Satzes ist die Summe der LinkWerte aller Links dieses Satzes.

3.1.3.6.1.8. Bestimmung des BildWertes

Bei der Bestimmung des Gewichtes eines Satzes wird nicht nur der direkte Inhalt des Satzes betrachtet. Wie im vorangegangenen Abschnitt gezeigt wurde, trägt auch der Inhalt der Zielseite eines Verweises zum Gewicht eines Satzes bei. Aber auch die Bilder, die nach einem Satz im Text abgebildet sind, ziehen wir zur Ermittlung des Satzgewichtes heran. Bilder spielen eine wichtige Rolle, da sie unterschiedlichste Sachverhalte veranschaulichen und erläutern können.
Jedoch trägt nicht jedes Bild zur Erhöhung des Gewichtes des vor ihm stehenden Satzes bei. Nur Bilder, die wichtige Informationen enthalten, sind hier von Interesse. Als wichtige Bilder werden alle die Bilder angesehen, die entweder in der Bildunterschrift oder im Namen des Bildspeicherortes bestimmte Schlüsselworte oder die vom Nutzer eingegebenen Suchworte enthalten. Solche Schlüsselworte sind zum Beispiel „Schema“, „Übersicht“, „Überblick“ oder „Veranschaulichung“. Da wir aber feststellten, dass oftmals Bilder nicht allein stehen, sondern in einen Satz integriert sind, wurde auch dieser Satz auf Vorkommen der oben erwähnten Worte durchsucht.
Damit der Informationsgehalt des Bildes aber nicht überschätzt wird, kann ein Satz für ein Bild maximal eine Gewichtserhöhung von 2 bekommen. Diese berechnet sich als Summe aus folgenden Summanden:

- $\begin{cases} 0.5; \text{ in Bildunterschrift ist mindestens ein Suchwort enthalten} \\ 0; \text{ sonst} \end{cases}$

- $\begin{cases} 0.5; \text{ in Bildunterschrift ist mindestens ein Schlüsselwort enthalten} \\ 0; \text{ sonst} \end{cases}$
- $\begin{cases} 0.5; \text{ in umgebendem Satz oder Speicherort ist mindestens ein Suchwort enthalten} \\ 0; \text{ sonst} \end{cases}$
- $\begin{cases} 0.5; \text{ in umgebendem Satz oder Speicherort ist mindestens ein Schlüsselwort enthalten} \\ 0; \text{ sonst} \end{cases}$

Diese Summe ist somit der BildWert eines Bildes. Die Summe der BildWerte aller Bilder, die direkt nach dem aktuellen Satz kommen, ist der BildWert dieses Satzes.

3.1.3.6.1.9. Bestimmung des SatzlängenWertes

Anfangs waren wir der Ansicht, dass jeder Satz einen SatzlängenWert erhalten sollte, der direkt von der Länge des Satzes abhängt, da in einen langen Satz viel mehr Informationen passen als in einen kurzen. Nach einigen Tests sahen wir allerdings, dass die Satzlänge selbst nicht direkt in die Berechnung eines Satzgewichtwertes eingehen sollte, da nicht alle langen Sätze auch gleichzeitig viele wichtige Informationen enthalten müssen. Gerade bei der Zusammenfassung von HTML-Seiten, bei denen die Tags für Style- und Formatangaben nicht entfernt wurden, erhielten so Sätze ein hohes Satzgewicht für Worte, die gar keinen Inhalt darstellen. Aber auch bei Sätzen, bei denen das nicht der Fall ist, steht die bloße Satzlänge nicht unbedingt für die Wichtigkeit des Satzes. Da wir aber der Meinung sind, dass sehr kurze Sätze kaum wichtige Inhalte enthalten, bekommen diese Sätze dafür ein negatives Gewicht von -5.

3.1.3.6.1.10. Bestimmung des WichtigeKonjunktionenWertes

Ein neuer Ansatz ist das Berücksichtigen von verschiedenen Konjunktionsarten. Da Konjunktionen Wörter, Wortgruppen und Sätze zum Zwecke der zeitlichen oder logischen Relativierung verbinden, haben sie einen erläuternden Signalwert, der etwas über die Art der Satz- bzw. Wortverbindung aussagt. Somit verknüpfen Konjunktionen Sätze zu Satzgefügen und fügen dabei einzelne Aussagen zu Gedanken zusammen. Da diese Satzgefüge mehr Inhalt als die einzelnen Aussagen aufweisen, weil sie diese in eine zeitliche oder logische Relation setzen, entschieden wir, dass diese Sätze wahrscheinlich wichtig für die Zusammenfassung sind. Allerdings wollten wir nicht alle Sätze mit Konjunktionen als potentielle Extraktionskandidaten ansehen, sondern nur diese, die mindestens ein Suchwort enthalten. Des Weiteren kamen wir zu dem Schluss, dass nur einige Konjunktionsarten mit logischer Bedeutung berücksichtigt werden: restriktive Konjunktionen (allein, nur, jedoch, sofern,

insofern, soweit, insoweit, außer), konditionale Konjunktionen(wenn, falls, sofern, soweit, solange), adversative Konjunktionen (aber, sondern, doch, wohingegen, während), kausale Konjunktionen (denn, da, weil, zumal), finale Konjunktionen (damit, auf dass) und konzessive Konjunktionen (obgleich, obwohl, obschon, obzwar, wenngleich, wenn auch, wennschon, wiewohl, ungeachtet).
Der WichtigeKonjunktionenWert, welcher nur für Sätze, in denen Suchworte enthalten sind, ermittelt wird, ist die Summe der im Satz enthaltenen Konjunktionen der vorhin erwähnten Konjunktionsarten.

3.1.3.6.1.11. Bestimmung des WichtigeÜberschriftenWertes

Bei diesem Wert wird ähnlich vorgegangen wie bei der Berechnung des Locationgewichtes nach Edmundson. Es wird die Überschrift des Abschnitts, in dem der zu bewertende Satz steht, auf das Vorhandensein von typischen Überschriftsworten, die auf die Wichtigkeit des Abschnittes hinweisen, durchsucht. Dazu werden aus einer Datei die Überschriftsschlüsselworte und die dazugehörigen Gewichte geladen und dann wird überprüft, welche von diesen Schlüsselworten in der zu untersuchenden Überschrift vorkommen. Der WichtigeÜberschriftenWert ist dann die Summe der Gewichte der in der Überschrift enthaltenen Schlüsselworte.

3.1.3.6.1.12. Bestimmung des FragenWertes

Wie Enrique Alfonseca und Pilar Rodríguez sind auch wir der Meinung, dass der Informationsgehalt von Fragen eher gering ist. Das Interessante und Wichtige an einer Frage ist ja bekanntlich deren Antwort. Aus diesem Grund berechnet sich der FragenWert wie folgt:

$$\text{FragenWert}(s) = \begin{cases} \text{wenn Satz eine Frage ist:} & -5 \\ \text{sonst:} & 0 \end{cases}$$

Abbildung 35: Berechnungsformel für den FragenWert

3.1.3.6.2. Auswahl der für die Zusammenfassung interessanten Bilder

Eine Besonderheit unseres Extraktionsalgorithmus ist die Berücksichtigung der Bilder bei der Berechnung des Satzgewichtes. Doch nicht nur an dieser Stelle finden die Bilder Beachtung. Auch in die Zusammenfassung sollen Bilder übernommen werden, welche für ein besseres Verständnis des Textes sorgen oder die zu den Nutzerinteressen passen. Um diese Bilder zu

identifizieren, werden der Speicherort eines Bildes, die Bildunterschrift und der Satz, in dem das Bild vorkommt, auf das Vorkommen von Schlüsselworten und Suchworten untersucht. Ein weiteres Kriterium, welches über die Übernahme eines Bildes in die Zusammenfassung entscheidet, ist der Abschnitt, in dem das Bild ursprünglich abgebildet war. Sollte aus diesem Abschnitt nicht mindestens ein Satz extrahiert worden sein, so wird auch kein Bild dieses Abschnittes für die Zusammenfassung extrahiert. Diese Entscheidung kann daher erst nach der Auswahl der wichtigsten Sätze stattfinden. Dazu wird als erstes in einem Integerfeld gespeichert, wie viel Sätze aus welchen Abschnitten für die Zusammenfassung selektiert wurden. Anschließend werden die Bilder in den AusgewählteBilder-Vektor übernommen, die mindestens ein Such- oder Schlüsselwort enthalten und aus deren Abschnitt mindestens ein Satz extrahiert wurde.

3.1.4. Generierung der Zusammenfassung

Da das Zusammenfassungssystem hauptsächlich für die Zusammenfassung von Internetseiten entwickelt wurde, entschieden wir uns die endgültige Zusammenfassung im HTML-Format zu erstellen.

Damit die Zusammenfassung einen möglichst informativen aber auch übersichtlichen Überblick über den Originaltext liefert, beschlossen wir, nicht nur die Überschriften von Absätzen einzufügen, von denen auch Sätze ausgewählt wurden, sondern von allen Absätzen. Dadurch bietet die Zusammenfassung einen recht vollständigen Einblick in den Inhalt des Originals. Wurden aus einem Absatz Sätze extrahiert, so stehen diese direkt nach der Überschrift. Wurde aber von den Sätzen eines Abschnittes keiner ausgewählt, so steht nach der Überschrift dieses Abschnittes unmittelbar die Überschrift des nächsten Abschnittes. Da bei Texten im RTF-Format öfters Abschnitte ohne direkte Überschrift vorkommen, weil hier bei der Unterteilung in Abschnitte nur nach Leerzeilen gesucht wird, und diese so die gleiche Überschrift wie der vorherige Abschnitt bekommen, mussten wir bei den Überschriften auch darauf achten, dass keine Überschrift mehrmals in die Zusammenfassung übernommen wird.

Bei der Erstellung der HTML-Zusammenfassung wird dabei Satzweise vorgegangen. Es werden die extrahierten Sätze, welche der Funktion erstelleZusammenfassungInHTML() in Form eines Integerfeldes übergeben wurden, nacheinander an einen String angehängt, welcher schließlich den kompletten Inhalt der HTML-Datei der Zusammenfassung enthalten soll. Dabei kann es eben sein, dass zwischen zwei extrahierten Sätzen noch Überschriften oder eventuell Bilder eingefügt werden.

Wurde nun eine Abschnittsüberschrift an den String, der letztendlich den gesamten Inhalt der Zusammenfassung enthalten wird, angehängt, werden bei Zusammenfassungen mit unserem eigenen Algorithmus die vorher als wichtig deklarierten Bilder beachtet. Aus dem Vektor, in dem die potentiell interessanten Bilder gespeichert wurden, werden die Bilder an dieser Stelle an den String angehängt, die aus dem gleichen Abschnitt sind und die im Original vor dem gerade betrachteten Satz standen. Dies ist möglich, da zu jedem Bild neben dem Speicherort und der Bildunterschrift auch die Id des Abschnittes, in dem es vorkam, und die Satzidentifikationsnummer des letzten Satzes vor dem Bild gespeichert wurden. Anschließend wird der Inhalt des aktuellen Satzes an den String angehangen und es wird überprüft, ob nach diesem Satz im Original ein Bild stand, welches nicht in den Vektor der wichtigen Bilder übernommen worden ist. Sollte ein solches Bild existieren, wird dieses auch an den schon erwähnten Zusammenfassungsstring angehangen, da wir der Meinung sind, dass dieses Bild auch wichtig sein kann. Es ist möglich, dass das im Satz Gesagte durch dieses Bild veranschaulicht werden soll und, da der Satz als wichtig eingestuft wurde und für die Zusammenfassung extrahiert wurde, ist es sehr wahrscheinlich, dass das darauf folgende Bild auch von Interesse ist. Danach wird überprüft, ob weitere Bilder aus dem Vektor, der die interessanten Bilder enthält, an diese Stelle eingefügt werden müssen. Dies ist nur dann der Fall, wenn Bilder existieren, die aus dem aktuellen Abschnitt stammen und die im Originaltext nach dem eben eingefügten Satz und vor dem nächsten extrahierten Satz standen. Sollten solche Bilder existierten, werden sie an den Zusammenfassungsstring angehangen.

Wurden auf diese Weise alle extrahierten Sätze und die als interessant deklarierten Bilder verarbeitet, werden noch Informationen über die Zusammenfassung eingefügt. Im letzten Abschnitt der Zusammenfassung kann so nachgelesen werden, wo der Originaltext gespeichert ist und wann diese Zusammenfassung mit welchem Komprimierungsgrad erstellt worden ist. Im Anschluss daran werden die für eine HTML-Datei notwendigen Strukturen hergestellt. Das heißt, es wird der eigentliche Inhalt, welcher im Zusammenfassungsstring gespeichert wurde, in die Body-Elemente eingeschlossen und es wird ein deklarativer Kopfabschnitt vorangestellt. Dieser durch die Head-Elemente begrenzte Abschnitt wird, wenn das Original ein HTML-Dokument war, aus dem Originaltext übernommen. Bei RTF-Dokumenten wird der Header aus den Textdaten erzeugt, das heißt, es werden zwischen die Title-Elemente die übergebenen Suchworte eingefügt.

Als letztes wird diese so erzeugte HTML-Datei an dem Speicherort, welcher über die GUI ausgewählt wurde, abgespeichert. Dabei kann diese Datei neu angelegt werden oder es wird eine schon bestehende Datei überschrieben.

4. Evaluation

Obwohl auf dem Gebiet der Automatischen Textzusammenfassung seit den 50er Jahren Forschung betrieben wird, konnte man in den letzten fünf Jahren einen raschen Anstieg der Arbeiten auf diesem Gebiet beobachten. Doch fast alle Forschungen wurden mit einem eigenen, internen Verfahren evaluiert. Dadurch ist es schwierig die Ergebnisse der verschiedenen Forschungen zu vergleichen, die Experimente zu wiederholen oder die Evaluationsdaten für Trainingszwecke weiter zu verwenden.
Auf dem Gebiet der Evaluation von Systemen zur Automatischen Textzusammenfassung sowie zur Entwicklung von einheitlichen, gemeinsam-nutzbaren Ressourcen fehlen anerkannte Standards.
Im Jahre 1998 fand erstmals eine Konferenz statt, welche die systematische Evaluierung von Textzusammenfassungssystemen zum Ziel hatte. Die Text Summarization Evaluation Conference bewertete im großen Rahmen und Entwickler-unabhängig die Brauchbarkeit der einzelnen Zusammenfassungssysteme in bestimmten Anwendungsbereichen.

4.1. Probleme bei der Evaluierung von Zusammenfassungen

Der fehlende Standard bei Evaluierungen von Zusammenfassungen ist darauf zurückzuführen, dass eine Reihe von Schwierigkeiten bei der Bewertung von Zusammenfassungen existiert.
Zum einen ist es nicht möglich, die eine richtige Zusammenfassung zu erstellen. Für jeden Text existiert eine unbegrenzte Anzahl an richtigen Zusammenfassungen. Selbst wenn es nur um eine extrahierende Zusammenfassung geht, bei der eine bestimmte Anzahl an vollständigen Sätzen aus dem Originaltext ausgewählt werden und zu einer Zusammenfassung zusammengefügt werden, wird es kaum zwei Menschen geben, die exakt die gleichen Sätze auswählen. Bei der Evaluation eines Zusammenfassungssystems, welches Extrakte mit Hilfe von genetischen Algorithmen erstellt, wollten Alfonseca und Rodríguez die Übereinstimmungen bei von Menschen erstellten Zusammenfassungen als Vergleichswert für die automatisch erstellten Zusammenfassungen messen. Sie gaben drei Testpersonen 10 Dokumente und ließen sie für drei verschiedenen Kompressionsraten und unterschiedliche Nutzerinteressen Multidokumentzusammenfassungen erstellen. Die Übereinstimmungen betrugen für je zwei Testpersonen zwischen 61.26 % und 72.13%. Dabei kann davon ausgegangen werden, dass bei der Erstellung allgemeiner Zusammenfassungen, die Übereinstimmungen noch geringer gewesen wären. Die Schwierigkeit, die sich daraus ergibt,

ist, dass wenn verschiedene Menschen unterschiedliche Zusammenfassungen erstellen, dann wäre es unangemessen von einem System zu erwarten, dass es eine Zusammenfassung erstellt, die sich mit denen von Menschen deckt. Dieses geringe Maß an Übereinstimmung zwischen verschiedenen Menschen stellt nicht nur bei der Erstellung von Referenzzusammenfassungen ein Problem dar sondern auch bei der Bewertung der Zusammenfassung. Denn auch bei der Beurteilung durch Testpersonen, ob die automatisch generierte Zusammenfassung der Referenzzusammenfassung in den einzelnen Inhaltspunkten entspricht, gibt es Differenzen zwischen den einzelnen Testpersonen und sogar zwischen ein und derselben Testperson, wenn sie dieselbe Zusammenfassung nach einer gewissen Zeit noch einmal beurteilen soll. Somit ist die Bewertung der Zusammenfassung durch Menschen nicht sehr zuverlässig.

Ein weiteres Problem ist die Originaltreue bei kritischen Zusammenfassungen, da diese Zusammenfassungen ja nicht nur relevante Informationen des Originals enthalten, sondern auch Meinungen über die Qualität und die Behauptungen des Textes. Außerdem kann ein automatisches System nur schwer die Qualität des Textes und dessen Behauptungen einschätzen.

Des Weiteren lassen sich Zusammenfassungen in verschiedenen Formen nur schwer vergleichen. Ist eine Zusammenfassung, welche nur aus Stichpunkten und Satzteilen besteht aber die relevanten Informationen enthält, genauso gut wie eine Zusammenfassung, die die relevanten Informationen in einem zusammenhängenden Text präsentiert?

Die größte Schwierigkeit bei der Evaluation von automatisch generierten Zusammenfassungen verschiedener Zusammenfassungssysteme ist das Messen der Performanz des Systems. Wie berechnet man die Qualität einer Zusammenfassung und des Zusammenfassungssystems? Welche Faktoren müssen berücksichtigt werden? Da sich Zusammenfassungen von verschiedenen Menschen unterscheiden, kann die Evaluation nicht auf den bloßen Vergleich mit einer von Menschen erstellten Zusammenfassung erfolgen.

4.2. Klassifizierung von Evaluierungsmethoden

Evaluierungsmethoden können nach unterschiedlichen Gesichtspunkten klassifiziert werden.

1. Intrinsisch ←→ Extrinsisch
   Intrinsische Evaluation bewertet direkt die Qualität der Zusammenfassung auf Grundlage von Analysen des Inhaltes der Zusammenfassung bezüglich eines Satzes von Regeln.

Extrinsische Evaluation beurteilt die Qualität der Zusammenfassung unter der Beachtung der Erfüllung bestimmter Aufgaben. Das heißt, es wird ermittelt, wie brauchbar die Zusammenfassung für die Erfüllung einer Aufgabenstellung ist. Solche Aufgaben können sein:

- Bewertung der Dokumentrelevanz
- Kategorisierung des Textes
- Beantwortung von Fragen
- Füllen von Templates

2. Black-Box ←→ Glass-Box

   Bei der Black-Box-Evaluierung wird das Gesamtsystem über Ein- und Ausgabe getestet. Bei Glass-Box-Evaluierung testet man die internen Module.

3. Offline ←→ Online

   Eine Offline-Evaluation kann von jedem System zu jeder gewünschten Zeit mit Hilfe eines automatischen Bewertungsprogramm durchgeführt werden. Für eine Online-Evaluation wird jemand benötigt, der das System testet.

4.2.1.Intrinsische Evaluierungsmethoden

Bei der Evaluation von Textzusammenfassungssystemen kommen meist intrinsische Evaluierungsmethoden, das heißt Evaluierungen, die auf Vergleichen mit einer „idealen“ Zusammenfassung, einer Menge von Zusammenfassungen oder dem Originaltext, beruhen, zum Einsatz. Ein Grund dafür, sind die geringeren Kosten gegenüber extrinsischen Verfahren. Intrinsische Verfahren bewerten dabei den Zusammenhang der Zusammenfassung sowie die Aussagefähigkeit.

4.2.1.1. Vergleich mit einer „idealen“ Zusammenfassung

Für die Erstellung der „idealen“ Zusammenfassung, auch Goldstandard genannt, welche als Vergleichsobjekt für die automatisch erstellten Zusammenfassungen dienen soll, gibt es unterschiedliche Ansätze. Zum einen kann sie durch einen professionellen Referenten erstellt werden. Zum anderen kann sie aus verschiedenen von Menschen erzeugten Zusammenfassungen entstehen. Dabei kann die Zusammenfassung entweder aus den Textteilen bestehen, welche die Mehrheit der Zusammenfassungen enthalten, oder sie entsteht durch die Vereinigung verschiedener Zusammenfassungen oder aber die Zusammenfassung

stellt den Durchschnitt der einzelnen von Menschen erzeugten Zusammenfassungen dar. Es ist aber auch möglich, eine Menge von „idealen“ Zusammenfassungen, welche von Menschen oder anderen Zusammenfassungssystemen erstellt wurden, für den Vergleich zu verwenden. Bei all diesen verschiedenen Möglichkeit besteht das gleiche Problem: Es gibt nicht die eine richtige Zusammenfassung und auch nicht eine endliche Menge richtiger Zusammenfassungen. Es besteht stets die Möglichkeit, dass ein Zusammenfassungssystem eine völlig andere Zusammenfassung erstellt, welche von der Aussagefähigkeit und dem Textzusammenhang genau so gut ist.
Durch eine einheitliche Länge der automatisch erstellten Zusammenfassungen und der Vergleichszusammenfassungen kann dieses Problem zwar verkleinert werden, aber es besteht weiterhin. Eine andere Möglichkeit zur Beschränkung dieses Problems, ist die alleinige Verwendung von automatisch erstellten Zusammenfassungen als Vergleichsobjekte. Da die meisten automatisch erstellten Zusammenfassungen extraktionsbasiert sind, bestehen sie alle aus Textteilen des Originaltexts. Da die Anzahl an solchen Textteilen im Originaldokument begrenzt ist, weisen diese Zusammenfassungen eine größere Ähnlichkeit untereinander auf als die von Menschen erstellten, welche oftmals komplett andere Textteile enthalten, die dieselbe Bedeutung wie das Original aufweisen.

4.2.1.1.1. Bewertung der Ähnlichkeit zur „idealen“ Zusammenfassung

Wenn das Problem der „idealen“ Zusammenfassung geklärt sein sollte, stellt sich die Frage: Was soll von wem bewertet werden, so dass als Ergebnis ein Wert für die Qualität der automatisch erzeugten Zusammenfassung vorliegt?
Die beste Möglichkeit zum Vergleichen von Zusammenfassungen ist dies von Menschen tun zu lassen, aber es besteht auch die Möglichkeit zur automatischen Berechnung der Qualität der Zusammenfassung.

4.2.1.1.1.1. Bewertung durch Menschen

Die Einschätzung des logischen Zusammenhalts und des Informationsgehalts einer Zusammenfassung sowie der Vergleich mit einer Referenzzusammenfassung kann von einem Menschen viel einfacher erledigt werden als von einem Programm. Ein Mensch kann beim Lesen sofort beurteilen, ob die Zusammenfassung zusammenhängend ist oder ob sie Lücken in der rhetorischen Struktur aufweist. Durch die Extraktion bestimmter Sätze und das Weglassen anderer Sätze kann es sein, dass das Bezugsobjekt eines Pronomens verloren geht und somit der Informationsgehalt dieses Satzes gegen Null geht. Des Weiteren kann ein

Mensch sofort feststellen, ob eine allgemeine Zusammenfassung die Hauptthemen des Originaltextes widerspiegelt. Außerdem ist ein Mensch in der Lage zu erkennen, wenn Teile des Textes mit Hilfe von Synonymen, Antonymen oder anderem Wortlaut mit gleicher Bedeutung wiedergegeben werden.

Evaluation mit SEE

Als Hilfsmittel für die Bewertung der Ähnlichkeit zwischen der Zusammenfassung und der Referenzzusammenfassung kann die Summary Evaluation Environment (SEE) von Chin-Yew Lin genutzt werden. Dabei werden die Zusammenfassung und der Referenztext in Einheiten (Sätze, Teilsätze, usw.) zerlegt. Während die Testperson beide Texte in den zwei Fenstern sehen kann, entscheidet sie für jede Unit der automatisch erstellten Zusammenfassung mit welcher Unit oder mit welchen Units des Referenztexts diese in Zusammenhang steht. Diese werden angeklickt und die Testperson entscheidet, ob die Unit in der Zusammenfassung die Unit im Referenztext völlig oder nur teilweise abdeckt und wie die grammatikalische Qualität der Unit ist. Als letztes können die Qualität, die Länge, die Inhaltsabdeckung, die Grammatik und die Organisation der automatisch erstellten Zusammenfassung eingeschätzt werden. Das System berechnet Recall und Precision für die Übereinstimmungen zwischen den beiden Texten. Nachteil dieser Methode ist allerdings, dass der Vergleich stets nur mit einer Referenzzusammenfassung möglich ist und nicht mit einer Sammlung von solchen.

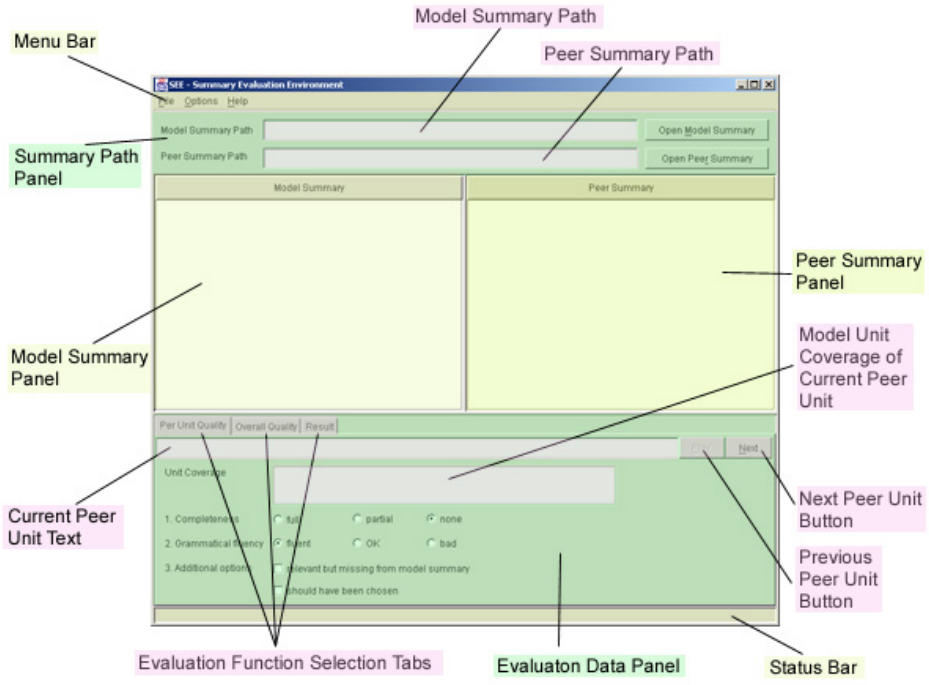


Abbildung 36: Oberfläche von SEE [Lin 2001, S. 4]

Bewertung von inhaltlicher Überdeckung

Um diesen Nachteil von nur einer Referenzzusammenfassung zu beheben versuchten van Halteren und Teufel [van Halteren & Teufel 2004] sowie Nenkova und Passonneau [Nenkova & Passonneau 2004] inhaltliche Basiseinheiten (auch Factoids oder SCUs genannt) durch den Vergleich verschiedener von Menschen erstellter Zusammenfassungen zu identifizieren. Van Halteren und Teufel zerlegten beispielsweise folgenden Satz „The police have arrested a white Dutch man“ anhand verschiedener Zusammenfassungen in die folgenden 5 Factoids:

- “A suspect was arrested”
- “The police did the arresting”
- “The suspect is white”
- “The suspect is Dutch”
- “The suspect is male” [van Halteren & Teufel 2004, S.3]

Bei jeder neuen Referenzzusammenfassung wurde nach schon identifizierten Factoids gesucht, welche eventuell aufgesplittet werden müssen, und es wurden stets auch neue entdeckt. Abhängig von der Anzahl ihres Auftretens wurden diese Factoids bei van Halteren und Teufel mit einem Relevanzgrad versehen. Der Goldstandard für dieses Verfahren war somit nicht eine Referenzzusammenfassung sondern all diese Basiseinheiten. Während der Evaluation mussten nun die Testpersonen die automatisch generierte Zusammenfassung in solche Basiseinheiten zerlegen, die korrespondierende Basiseinheit aus dem Goldstandard auswählen und deren Relevanzwert übernehmen. Somit ergab sich die Relevanz der Zusammenfassung aus der Summe der Relevanzwerte für die Basiseinheiten.

Bei Nenkova und Passonneau wurden die Sätze in SCUs zerlegt, welche anschließend abhängig von ihrem Aufkommen gewichtet und in eine Pyramide eingeordnet wurden. Die folgenden vier Sätze aus vier verschiedenen Zusammenfassungen für den gleichen Text und ihre SCUs veranschaulichen die Vorgehensweise:

- “In 1998 two Libyans indicted in 1991 for the Lockerbie bombing were still in Libya.”
- “Two Libyans were indicted in 1991 for blowing up a Pan Am jumbo jet over Lockerbie, Scotland in 1988.”
- “Two Libyans, accused by the United States and Britain of bombing a New York bound Pan Am jet over Lockerbie, Scotland in 1988, killing 270 people, for 10 years were harbored by Libya who claimed the suspects could not get a fair trail in America or Britain.”
- “Two Libyan suspects were indicted in 1991.” [Nenkova & Passonneau 2004, S. 2]

Aus diesen Sätzen ergab sich für die SCU1 „two Libyans were officially accused of the Lockerbie bombing“ ein Gewicht von 4, da sie in allen vier Sätzen zu finden war. Für die SCU2 „the indictment of the two Lockerbie suspects was in 1991” konnte nur ein Gewicht von 3 ermittelt werden, da es im dritten Satz fehlt. Die zu bewertende Zusammenfassung wurde dann anhand der fertig konstruierten Pyramide bewertet. Der Relevanzwert der Zusammenfassung berechnet sich dann als das Verhältnis zwischen der Summe der Gewichte der SCUs der Zusammenfassung und der Summe der Gewichte der SCUs einer optimalen Zusammenfassung mit gleicher Anzahl von SCUs.
Der Vorteil dieser Methoden liegt zum einen darin, dass nicht eine einzelne Zusammenfassung als Vergleichsobjekt dient, und zum anderen darin, dass der Vergleich nicht auf Stringvergleiche zurückgeführt wird sondern auf Inhaltsvergleiche.

Evaluation unter Berücksichtigung des relativen Nutzens
Mit dieser Methode, welche von D. R. Radev und D. Tam [Radev & Tam 2003] entwickelt wurde, können trotz niedriger Übereinstimmung bei der Beurteilung durch verschiedene Testpersonen gute Ergebnisse erzielt werden. Die Referenzzusammenfassungen werden von den Testpersonen auf folgende Art und Weise erstellt: die Testpersonen bewerten alle Sätze des Originaltextes entsprechend ihrer Relevanz und aus dieser Rangfolge werden dann die x bestbewerteten Sätze für eine Zusammenfassung mit der Länge x ausgewählt. Selbst wenn zwei Testpersonen vollkommen unterschiedliche Zusammenfassungen erstellt haben, kann man aus den Relevanzwerten für die ausgewählten und nicht ausgewählten Sätze beider Testpersonen ein Übereinstimmungsmaß berechnen. Auf Grundlage der Relevanzwerte und daraus berechneter Werte wie Interjudge Agreement, Random Performance und Systemperformanz wurden die automatisch erstellten Zusammenfassungen von vier verschiedenen Systemen (MEAD, LEAD, RANDOM, WEBSUM) bewertet. Es zeigte sich, dass wesentlich sinnvollere Ergebnisse erzielt werden konnten als bei Precision und Recall.

Allerdings gibt es einige Probleme bei der Evaluation von Zusammenfassungssystemen mit Hilfe von Menschen. Ein Problem ist, dass eine solche Evaluation stets eine gewisse Planungsphase braucht, in welcher Testpersonen gefunden werden müssen und die Evaluation vorbereitet wird. Dann dauert die Evaluation der Texte eine gewisse Zeit, da die Menschen sowohl die erstellte Zusammenfassung als auch die Menge der Referenzzusammenfassungen mehrmals durchlesen müssen, um sie im Anschluss bewerten zu können. Somit kann eine Evaluation nicht zu jeder Zeit durchgeführt werden. Das alles macht diese Art von Evaluation sehr kostspielig.

Ein weiteres Problem bei der Bewertung durch Menschen ist die fehlende Objektivität. Viele Studien haben gezeigt, dass die Menschen bei der Entscheidung, welche Sätze für eine Zusammenfassung relevant sind, zu unterschiedlichen Ergebnissen kommen. Rath, Resnick und Savage zeigten in einer Studie, welche das Erstellen von 20-zeiligen Extrakten von 10 Artikeln durch Menschen mit sich brachte, dass die 6 Testpersonen pro Extrakt nur in 1.6 Sätzen übereinstimmten und dass ein und dieselbe Person acht Wochen später einen wesentlich anderen Extrakt vom gleichen Artikel produzierte. Auch Salton, Singhal, Mitra und Buckley machten ähnliche Erfahrungen. Bei der Auswahl von 5 Paragraphen von je 50 Artikeln erreichten die beiden Testpersonen lediglich eine Übereinstimmung von 46 %. Im Gegensatz dazu konnten Jing, Barziley, McKeown und Elhadad Übereinstimmungen bis zu 96 % Prozent unter 5 Testpersonen ermitteln, welche Extrakte mit 10 % Länge des Originalstexts erstellten sollten, und Übereinstimmungen bis 90 % bei Extrakten mit 20 % Länge. Einen Grund dafür sahen Jing, Barziley, McKeown und Elhadad in der einheitlichen Struktur der Texte. Jeder dieser Artikel stammte von der Text Retrieval Conference (TREC) und begann mit einer Einleitung, welche eine sehr gute Zusammenfassung war, gefolgt von Details, Beispielen, Fakten und so weiter.

.

4.2.1.1.1.2. Automatische Bewertung

Zur automatischen Bewertung der Ähnlichkeit zwischen der automatisch generierten Zusammenfassung und der „idealen“ Zusammenfassung gibt es verschiedene Messverfahren:

- *Sentence Recall und Precision*: Beim Recall-Wert wird gemessen wie viele Sätze aus der „idealen“ Zusammenfassung auch in automatisch generierten enthalten sind. Precision gibt den Anteil der relevanten Sätze unter den für die Zusammenfassung ausgewählten an. Precision und Recall sind Standardmesswerte aus dem Information Retrieval Bereich. Dieses Verfahren kann hauptsächlich für Zusammenfassungen, welche durch Satzextraktion entstehen, verwendet werden. Problematisch ist allerdings, dass man mit diesem Verfahren nicht zwischen verschiedenen, gleich guten Zusammenfassungen differenzieren kann und dass Zusammenfassungen, die sich vom Inhalt her stark unterscheiden, ähnliche Werte erhalten können.
- *Sentence Rank*: Bei diesem Verfahren werden die automatisch generierte und die „ideale“ Zusammenfassung als eine Rangliste von Sätzen angegeben. Die Sätze erhalten ihren Rang in Hinblick auf ihren Wert für die Zusammenfassung. Diese Satzordnungen werden dann mit einer Übereinstimmungsmessung verglichen. Auch dieses Verfahren ist nur für Extrakte und nicht für Abstrakte sinnvoll einsetzbar.

Außerdem ist es für Menschen eine unnatürliche Vorgehensweise für die Erstellung von Zusammenfassungen.

- *String-Überdeckung*: Laut Lin und Hovy korreliert dieses Verfahren mit der Vorgehensweise von Menschen beim Vergleichen von Zusammenfassungen. Es wird dabei n-gram-basiert nach Überdeckungen in der automatisch erstellten und der „idealen" Zusammenfassung gesucht. Allerdings kann die bloße Suche nach identischen Strings nicht gewährleisten, dass auch die gleiche Bedeutung vorhanden ist und keine Negation erfolgt. Dies zeigt folgendes Beispiel:

  Original:„Polizei erschoss Geiselnehmer"
  Zusammenfassung: „Geiselnehmer erschoss Polizei"
  → dies müsste eine gute Zusammenfassung sein, da alle Strings enthalten sind

  Für Extrakte, Abstrakte mit einem hohen Gehalt an „Copy-and-Paste" des Originals oder für Zusammenfassungen in Form von Stichwortlisten liefert dieses Verfahren dennoch brauchbare Ergebnisse.
- *Inhaltsüberdeckung*: Bei dieser Methode wird die Referenzzusammenfassung in Key Concepts zerlegt und mit Nutzwerten versehen. Auch die automatisch erstellte Zusammenfassung wird in solche Einheiten zerlegt. Anschließend wird automatisch überprüft, welche Einheiten aus der automatisch erstellten Zusammenfassung mit den Referenzeinheiten korrespondieren, und dann werden die entsprechenden Nutzwerte übertragen. Die Summe aller Nutzwerte ergibt die Qualität der Zusammenfassung. Amigó et al. [Amigó, Gonzala, Peinado, Penas & Verdejo 2004] haben dieses ursprünglich nur manuell durchführbare Verfahren automatisiert. Durch dieses Nutzwert-basierte Verfahren ist eine präzisere Messung des Informationsgehaltes möglich als bei einfacher boolescher Bewertung.
- *Automation of Pyramid method:* A. Harnly automatisierte einen Teil der Pyramid Methode, welche von A. Nenkova und R. Passonneau entwickelt wurde (siehe Kapitel 4.2.1.1.1.1.). Während das Erstellen der Pyramide mit den SCU-Gewichten noch „per Hand" geschehen muss, geschieht die Bewertung der eigentlichen Zusammenfassung automatisch. Der Algorithmus zerlegt jeden Satz in alle möglichen SCUs, vergleicht diese mit den SCUs in der Pyramide und wählt dann die Menge der SCUs so, dass alles abgedeckt wird und diese Menge disjunkt ist. Anhand dieser Menge erfolgt letztendlich die Bewertung mit Hilfe der Gewichte der Pyramide.

Der Vorteil von automatischen Bewertungen ist, dass sie zu jeder Zeit durchgeführt werden können. Denn existiert eine Referenzzusammenfassung beziehungsweise eine Menge von Referenzzusammenfassungen, so kann nach jeder Änderung und Erweiterung des Zusammenfassungssystems eine Evaluation durchgeführt werden. Allerdings sind durch automatische Bewertung Aussagen über die Lesbarkeit, Fehler in der Grammatik oder Verwüstungen von Strukturen wie Listen und Tabellen nur schwer realisierbar.

4.2.1.2. Vergleich mit dem Originaltext

Um das Problem mit dem Finden von Referenzzusammenfassungen zu umgehen, kann bei der Evaluation die erstellte Zusammenfassung auch mit dem Originaltext verglichen werden. Zum Beispiel kann das Verfahren des Sentence Rank ebenso auf den Originaltext angewendet werden, wobei die Testperson alle Sätze des Originals entsprechend ihrer Relevanz ordnet und diese Rangliste mit den Sätzen der Zusammenfassung vergleicht. Eine andere Möglichkeit fanden H. Saggion und G. Lapalme, welche anhand des Originaltextes Stichwortlisten anfertigen ließen. Die Testpersonen erhielten die automatisch generierte Zusammenfassung und verschiedene Stichwortlisten. Anhand der Zusammenfassung sollte nun die richtige Stichwortliste zugeordnet werden. Wenn dies möglich war, so gingen sie davon aus, dass die Zusammenfassung die Hauptthemen des Originals enthielt. [Saggion & Lapalme 2000]

Eine weitere Möglichkeit der Evaluation von automatisch erstellten Zusammenfassungen anhand des Originaltextes liefern C. D. Paice und P. A. Jones. [Paice & Jones 1993]. Sie charakterisieren Ausdrücke im Originaltext als fokal, nicht fokal und als Schlussfolgerung. Fokal sind dabei Ausdrücke die grundlegend und unerlässlich für den Originaltext sind, nicht fokal sind hingegen alle anderen relevanten Ausdrücke. Die Überdeckung der Zusammenfassung mit diesen Ausdrücken wurde dann bewertet als korrekt, nicht korrekt und fehlend. Die daraus resultierende Statistik soll die Leistung des Zusammenfassungssystems charakterisieren.

4.2.2. Extrinsische Evaluierungsmethoden

Die zweite Klasse von Evaluierungsmethoden ist die extrinsische Evaluation. Diese Evaluierungsmethoden sollen herausfinden, wie gut die automatisch erstellten

Zusammenfassungen Menschen bei der Erfüllung einer bestimmten Aufgabe unterstützen. Solche Aufgaben können sein:

- Anhand der Zusammenfassung die Relevanz des Originaldokuments bezüglich eines bestimmten Ereignisses oder einer bestimmten Thematik einschätzen (Kategorisierungs-Aufgaben)
- Frage-Antwort-Aufgaben
- Ist es möglich Instruktionen, die in Zusammenfassung enthalten sind, korrekt auszuführen (Leseverständnis-Aufgaben)
- Anhand der Zusammenfassung aus einer großen Sammlung von Dokumenten das richtige auswählen, um einen effektiven Report oder eine Präsentation erstellen zu können (Retrieval-Aufgabe)
- Die Zusammenfassung in eine akzeptable, für eine Aufgabe benötigte Form bringen, um so den Arbeitsaufwand für die Nachbearbeitung zu bestimmen
- Rekonstruktion des Originaltextes anhand der Zusammenfassung (Shannon-Aufgabe)

4.2.2.1. Kategorisierungs-Aufgabe (identification task)

In der Kategorisierungs- oder Relevanzeinschätzungsaufgabe erhält die Testperson die automatisch generierte Zusammenfassung sowie eine Thematik und soll unter Verwendung der Zusammenfassung die Relevanz des Originaltextes in Bezug auf die gegebene Thematik einschätzen. Es kann auch sein, dass die Testperson verschiedene Themenbereiche erhält und den Text anhand der Zusammenfassung zu dem passenden Themenbereich zuordnen soll. Anhand der Genauigkeit und der dafür benötigten Zeit dieser Kategorisierung oder der getroffenen Relevanzentscheidung wird die Bewertung der Zusammenfassung vorgenommen.
Bei der TIPSTER SUMMAC Evaluation wurde überprüft, ob eine allgemeine Zusammenfassung genug Informationen über die Hauptthemen des Originals enthält, so dass man schnell und korrekt das Dokument kategorisieren kann. Dabei sollten die Testpersonen den erhaltenen Text, welcher die automatisch erstellte Zusammenfassung oder der Originaltext sein konnte, in eine von fünf Kategorien einordnen oder entscheiden, dass der Text in keine dieser Kategorien passt. So wurde entschieden, dass eine Zusammenfassung fehlerfrei war, wenn mit ihrer Hilfe die Relevanz oder Irrelevanz des Originals hinsichtlich einer bestimmten Thematik präzise bestimmt werden konnte. Anhand der gemessenen Zeit, die für die Kategorisierung mit Hilfe der Zusammenfassungen nötig war, konnte gezeigt werden, dass die Relevanzentscheidungen durch die automatisch generierten

Zusammenfassungen mit einer Zeitersparnis zwischen 40% und 50% getroffen wurden, ohne bemerkenswerte Verluste in der Präzision.

Auch Mani und Bloedorn [Mani & Bloedorn 1999] nutzen die Kategorisierungsaufgabe als extrinsische Evaluierungsmethode. Sie präsentierten den Testpersonen, welche dachten, sie nähmen an einer Information Retrieval Forschung teil, stets eine Anfrage und einen Text. In einem Testdurchlauf war der Text das Originaldokument und im anderen die Zusammenfassung, welche aus den fünf Sätzen mit den höchsten Gewichten bestand. Die Testpersonen erhielten in keinem der beiden Durchläufe den gleichen Text mit der gleichen Anfrage. Für jede Kombination aus Text und Anfrage musste entschieden werden, ob der Text für diese Anfrage relevant oder irrelevant ist. Die Anfragen sowie die Dokumente wurden aus der TREC-Sammlung ausgewählt. Die Messung der Genauigkeit der getroffenen Entscheidungen wurde über Recall und Precision ermittelt.

Des Weiteren nutzen Tombros und Sanderson [Tombros & Sanderson 1998] die Evaluation über die Kategorisierungsaufgabe, um die Vorteile ihrer auf eine Anfrage ausgerichteten, automatisch erstellten Zusammenfassungen zu beweisen. Sie kombinierten dafür ihr Zusammenfassungssystem mit einem Information-Retrieval-System so, dass für eine Anfrage eine nach Relevanz geordnete Liste mit Überschriften, Zusammenfassungen als Vorschau für den Originaltext und Links zu den Originaldokumenten präsentiert wurden. Sie verwendeten für einen Versuchsdurchlauf die auf eine Anfrage ausgerichteten, automatisch erstellten Zusammenfassungen und für einen weiteren Zusammenfassungen, die aus dem Titel und den ersten paar Sätzen des Originals bestanden. Die Aufgabe der Testpersonen war es, in 5 Minuten so viele Dokumente wie möglich bezüglich ihrer Relevanz für die Anfrage zu bewerten. Dazu konnten sie die Zusammenfassung und das Originaldokument nutzen. Ausgewertet wurden im Anschluss Recall und Precision der Relevanzentscheidungen, die dafür benötigte Zeit, ob es nötig war, das Originaldokument zu Rate zu ziehen, und die subjektive Meinung der Testpersonen bezüglich der Arbeitserleichterung durch die Zusammenfassungen.

4.2.2.2. Leseverständnis-Aufgabe und Frage-Antwort-Aufgabe

Meist versteht man unter der Leseverständnis-Aufgabe dasselbe wie unter der Frage-Antwort-Aufgabe. Dies ist nachvollziehbar, da Fragen zum Text nur dann richtig beantwortet werden können, wenn man den Inhalt der Zusammenfassung verstehen kann. Allerdings kann das Leseverständnis auch mit anderen Mitteln, wie zum Beispiel dem Ausführen von im Text

enthaltenen Instruktionen (beispielsweise bei Bauanleitungen oder Rezepten) überprüft werden.

Diese Art der Evaluation läuft in zwei Stufen ab, wobei in der ersten Stufe Fragen erstellt werden, die mit dem Originaltext beantwortet werden können. Für den zweiten Teil dieser Evaluationsmethode lesen Testpersonen entweder die Zusammenfassung oder den kompletten Text von einem oder mehreren Dokumenten. Anschließend müssen sie meist Fragen beantworten oder einen Multiple-Choice-Test ausfüllen. Der Vorteil von Multiple-Choice-Test ist, dass das System automatisch die Punkte der Antworten berechnen kann. Können die Fragen genauso korrekt beantwortet werden, wie das mit dem Original möglich wäre, so wird davon ausgegangen, dass die Zusammenfassung sehr aussagefähig und informativ ist.

Beispielsweise versuchten Morris, Kasper und Adams [Morris, Kasper & Adams 1992] die Bedeutung von Zusammenfassungen mit einer Frage-Antwort-Aufgabe nachzuweisen. Sie wählten dafür 4 Leseverständnisübungen von GMAT, einem standardisierten Test zur Beurteilung der Eignung für betriebswirtschaftliche Studiengänge. Um das Leseverständnis zu überprüfen, mussten die Testpersonen nach dem Lesen des Textes einen Multiple-Choice-Test absolvieren. Dieser Test wurde mit unterschiedlichen Texten durchgeführt: mit dem Originaltext, mit automatisch generierten allgemeinen Extrakten, mit von Menschen erstellten informativen Zusammenfassungen und ohne Text.

Ebenfalls Maybury [Maybury 1995] nutzte diese Art der Evaluation um das SUMGEN-Zusammenfassungssystem zu evaluieren. SUMGEN erstellt Zusammenfassungen von zwei verschiedenen Quellen: die Ausgaben einer Kampfsimulation und Business-Nachrichten über Arbeitsgemeinschaften, welche aus Templates mit Hauptinformationen gebildet wurden. Für beide Quellen und deren Zusammenfassungen mussten Testpersonen Fragen beantworten. Hierbei handelte es sich nicht um Multiple-Choice-Fragen, sondern für die Kriegssimulation mussten Fragen nach dem Namen, den Teilnehmern, der Zeit sowie der Dauer aller im Text genannten Missionen beantwortet werden und bei den Business-Nachrichten wurde nach dem Typ, den Partnern sowie dem Status der Arbeitsgemeinschaft im Text gefragt.

Da die Evaluation über Frage-Antwort-Aufgaben teuer ist und eine gewisse Anzahl an Testpersonen benötigt wird, haben sich Hirao, Okumura, Fukushima und Nanba [Hirao, Okumura, Fukushima & Nanba 2004] für eine Pseudo-Frage-Antwort-Aufgabe entschieden, welche ohne eine Beantwortung von Fragen durch Menschen auskommt. Für jedes Dokument wurden 5 Fragen für kurze Zusammenfassungen und 10 Fragen für lange Zusammenfassungen verfasst. Dann wurde jede Zusammenfassung nach ihrem „Exact-Match“, eine Bewertungsfunktion, welche 1 zurückliefert, wenn die Antwort auf die Frage in

der Zusammenfassung enthalten ist, und ihrer „Edit-Distance“, ein Messwert, der angibt, ob die Zusammenfassung Strings enthält, die ähnlich dem Antwortstring sind, bewertet.

4.2.2.3. Shannon-Aufgabe

Die Shannon-Aufgabe ist eine Ableitung des Shannon-Maßes, welches in der Informationstheorie Anwendung findet. Bei dieser Art der extrinsischen Evaluation sollen die Testpersonen den Originaltext, von dem sie entweder die Zusammenfassung oder das Originaldokument oder gar nichts gesehen haben, rekonstruieren. Bei dieser Aufgabe wird versucht den Informationsinhalt dadurch zu quantifizieren, dass das nächste Zeichen oder Wort erraten werden soll. Die Informationszurückbehaltung wird dann in der Anzahl der Tastenanschläge ermittelt, welche nötig waren, um die Originalpassage zu rekonstruieren.
Ein Nachteil dieser Methode ist allerdings, dass die Ergebnisse sehr stark von der Testperson abhängen. Die zu erwartenden Messwerte können sich durch das Wissen der Testperson bezüglich der Sprache und der Domäne stark verändern.
Hovy und Marcu [Hovy & Marcu 1998] nutzten die Shannon-Aufgabe zur Evaluierung. Hierbei bekam eine erste Gruppe von Testpersonen den Originaltext, einen Nachrichtenartikel der LA Times, um ihn langsam und gründlich zu lesen. Dann musste diese Gruppe, ohne den Originaltext zu sehen, das Original Buchstabe für Buchstabe rekonstruieren. Bei einem falsch geratenen Buchstaben musste noch einmal geraten werden, bei einem richtig geratenen ging es zum nächsten Buchstaben. Die Punktzahl der Testpersonen ergab sich aus der Anzahl falsch geratener Buchstaben geteilt durch die Gesamtanzahl an zu erratenen Buchstaben. Für die zweite Gruppe lief das Experiment genauso ab, aber mit dem Unterschied, dass sie nicht den Originaltext sondern ein 300-Wörter-lange Zusammenfassung des Nachrichtenartikels lesen mussten. Die dritte Gruppe fing gleich mit dem Raten der Buchstaben an, ohne irgendetwas gelesen zu haben und somit auch ohne Wissen über die Thematik des Textes. Die erreichten Punktzahlen der ersten und der dritten Gruppe dienten als oberer und unterer Grenzwert. Umso kleiner der Unterschied zwischen der Punktzahl der zweiten Gruppe und dem oberen Grenzwert war, umso besser war die Zusammenfassung. Während die erste Gruppe durchschnittlich 11 Fehler und die zweite Gruppe durchschnittlich 150 Fehler machte, waren 1100 Fehler der Durchschnittswert für die dritte Gruppe, welche den Text selbst nach 3 Stunden noch nicht rekonstruiert hatten.

4.2.2.4. Retrieval-Aufgabe

Bei der Evaluierung durch die Retrieval-Aufgabe, auch Ad hoc – Aufgabe genannt, wird das Zusammenfassungssystem als Back-End für ein Information-Retrieval-System betrachtet. Dabei erhält der Nutzer als Ergebnis für die von ihm gestellte Anfrage nicht den kompletten Text sondern eine auf diese Anfrage zugeschnittene Zusammenfassung. Anhand der Zusammenfassung soll der Nutzer dann die Relevanz des Originaltextes für seine Anfragen einschätzen können. Bei der Evaluierung werden dann die Zeit und die Genauigkeit gemessen. Diese Evaluationsmethode ist für die Bewertung nutzerorientierter oder Anfrage-abhängiger Zusammenfassungen besonders geeignet.

Jing, Barzilay, McKeown und Elhadad [Jing, Barzilay, McKeown & Elhadad 1998] untersuchten auch diese Evaluationsmethode. Allerdings hatten sie keine Möglichkeit nutzerorientierte Zusammenfassungen automatisch generieren zulassen. Sie behalfen sich damit, dass sie allgemeine Zusammenfassungen von Texten, deren Hauptthema zur ausgewählten Abfrage passte, automatisch erstellen ließen. Die aus der TREC-Sammlung ausgewählten Abfragen waren:

- „Status of nuclear proliferation treaties – violations and monitoring“
- “What evidence is there of paramilitary activity in the U.S.?”
- “What are the different techniques used to create self-induced hypnosis?”
- “What was responsible for the great emergence of ‘MICROSOFT’ in the computer industry?”

Für jede dieser Anfragen wurden 10 Texte ausgewählt. Für jeden dieser Texte wurden 11 Formen erstellt:

- Originaltext
- Schlüsselworte
- Titel
- Zusammenfassungen von 10% und 20% Länge von 3 verschiedenen Zusammenfassungssystemen
- Zusammenfassungen von 10% und 20% Länge von Menschen erstellt

Die drei Gruppen zu je vier Leuten erhielten 4 Anfragen und 40 Dokumente, von denen jeweils 10 Dokumente zu einer bestimmten Anfrage gehörten und in der gleichen Form vorlagen. Die Testpersonen musste die Relevanz der Dokumente für die jeweilige Anfrage bestimmen und dabei die dafür benötigte Zeit notieren. Die Werte für die Schlüsselworte und für die von Menschen erstellten Zusammenfassungen wurden als Basiswerte betrachtet.

Auch in der TIPSTER SUMMAC Evaluation wurde mit der Retrieval-Aufgabe evaluiert. Dazu wurden 20 verschiedene Themen aus der TREC-Sammlung ausgewählt und für jedes dieser Themen aus den 200 relevantesten Texten, die von einem Information Retrieval System zurückgeliefert wurden, 50 Dokumente ausgewählt. Die Dokumentmengen wurden so konstruiert, dass 25% bis 75% der Dokumente relevant für das Thema waren. Im ersten Experiment sollten die Ergebnisse für Zeitbedarf und Genauigkeit der unterschiedlichen Testbedingungen (kompletter Text, Zusammenfassung mit vorgegebener Länge von 10% des Originals, Zusammenfassung mit variabler Länge, Zusammenfassung aus den ersten 10% des Originals) erforscht werden. Bei den Zusammenfassungen mit fester und variabler Länge handelte es sich um indikative, nutzerorientierte Zusammenfassungen. Während im zweiten Experiment die Performanz verschiedener Zusammenfassungssysteme beurteilt wurde.
Die 21 Testpersonen des ersten Experiments sollten entscheiden, ob ein Text, welcher sowohl das Original als auch eine Zusammenfassung sein konnte, für das angegebene Thema relevant ist oder nicht. Es sollte damit geklärt werden, ob Anfragen-relevante Zusammenfassungen effektiv die Relevanz des Originals anzeigen können. Die Genauigkeitsperformanz wurde mit Precision, Recall und dem daraus berechenbaren F-Score beurteilt. Mit Hilfe dieser Werte konnte berechnet werden, dass bei der Verwendung von Zusammenfassungen mit variabler Länge die Relevanzentscheidungen doppelt so schnell und mit einem geringen Genauigkeitsverlust von 4% getroffen werden können (vgl. [Mani, Klein, House & Hirschman 2001]).

4.2.3. Vergleich intrinsischer und extrinsischer Evaluierungsmethoden

Sowohl intrinsische als auch extrinsische Evaluierungen haben ihre Vor- und Nachteile. Zum Beispiel ist der Vorteil von intrinsischen Evaluationen, dass sie bereits in frühen und mittleren Entwicklungsstufen des Zusammenfassungssystems eingesetzt werden können. Im Gegensatz dazu können extrinsische Methoden erst auf Systemebene durchgeführt werden, wenn das System komplett entwickelt ist. Nachteile intrinsischer Evaluation sind allerdings die Beschränkung auf ein oder wenige Referenzzusammenfassungen sowie die geringe Übereinstimmung zwischen verschiedenen menschlichen Bewertern. Aber auch die bloße Beurteilung eines Zusammenfassungssystems nach Precision und Recall, wie das bei intrinsischen Verfahren üblich ist, ist als Nachteil zu betrachten, da diese Werte keine wirklichen Aussagen über die Qualität einer Zusammenfassung liefern können. Allerdings muss erwähnt werden, dass die intrinsische Methode wesentlich kostengünstiger und daher

weiter verbreitet ist. Für die weite Verbreitung dieser Evaluierungsmethode spricht auch die Möglichkeit, diese Evaluation zu automatisieren. Gerade bei der Einschätzung von automatisch generierten Extrakten kann über automatische Messungen wie Sentence Recall und Vokabeltest auf einfache und preiswerte Weise vorgegangen werden. Im Unterschied dazu können extrinsische Evaluierungsverfahren nur semi-automatisch durchgeführt werden. Das heißt, es müssen menschliche Testpersonen die Zusammenfassungen in unterschiedlichen Aufgaben testen. Die dabei erzielten Ergebnisse können automatisch ausgewertet werden. Durch die Auswertung durch Testpersonen wird die extrinsische Evaluation langsamer und somit auch teuerer.

Ein weiterer Nachteil der extrinsischen Evaluation kann laut Laura Alonso i Alemany darin gesehen werden, dass durch die Ausrichtung auf die Anwendungsaufgabe die wirklichen Erkenntnisse über die Zusammenfassung verschleiert und verdunkelt werden (vgl. [Alonso i Alemany 2005]). Gerade in dieser Möglichkeit, durch eine gezielte Anwendungsaufgabe die Zusammenfassung in der Umgebung und unter den Bedingungen zu testen, in der sie später genutzt werden soll, sehen McLellan, Tombros, Jose, Ounis und Whitehead [McLellan, Tombros, Jose, Ounis & Whitehead 2001] aber einen großen Vorteil der extrinsischen Evaluation.

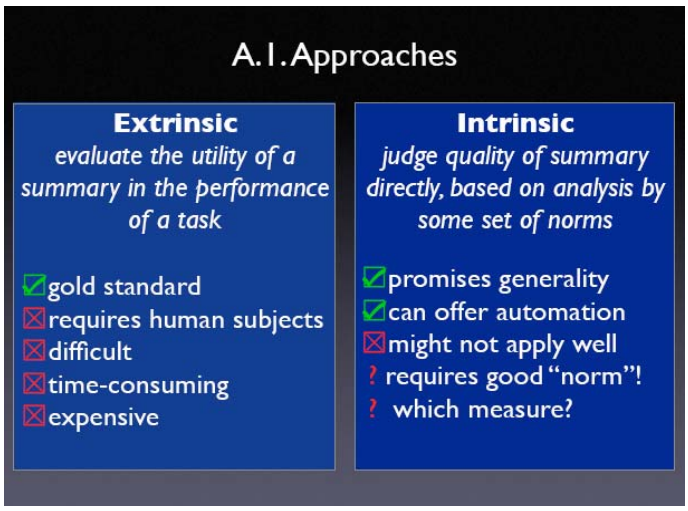


Abbildung 37: Gegenüberstellung intrinsische und extrinsische Evaluation [Harnly 2005]

4.2.3. Versuche zur Standardisierung der Evaluation

Da die Möglichkeiten zu Evaluation von Textzusammenfassungssystemen relativ vielfältig sind und die Ergebnisse verschiedener Evaluationen von unterschiedlichen Zusammenfassungssystemen nicht vergleichbar sind, versuchten sowohl die Document

Understandig Conference als auch die amerikanische Regierung mit der TIPSTER Text Summarization Evaluation (SUMMAC) Möglichkeiten zur Standardisierung der Evaluationen von Textzusammenfassungssystemen zu finden.

4.2.3.1. Document Understandig Conference

Durch das große Interesse und die vielen Aktivitäten im Bereich der Entwicklung von vielseitigen Informationssystemen von DARPA, ARDA und NIST wurden für die unterschiedlichen Aufgabenbereiche auch unterschiedliche Evaluationsmethoden nötig. Ein wichtiger Meilenstein dafür war das TIDES-Programm von DARPA, einer Pentagon-Agentur, die Hightech-Projekte für das amerikanische Militär durchführt. Bei diesem Translingual Information Detection, Extraction and Summarization Programm (TIDES) ging es um eine erweiterte Sprachverarbeitungstechnologie, die es Englisch sprechenden Menschen ermöglicht, in anderssprachigen Texten kritische Informationen zu finden und zu interpretieren ohne Wissen über die jeweilige Sprache. Einige Forscher aus diesem Programm bildeten mit weiteren Interessenten eine Gruppe, welche sich auf automatische Textzusammenfassung und deren Evaluation spezialisierte.
Im Herbst 2000 fand erstmals ein zweitägiger Workshop statt, bei dem es um unterschiedliche Methoden zur Evaluierung von Zusammenfassungen ging. Aus diesem Workshop und weiteren Bemühungen um eine langfristige Evaluation im Bereich der Zusammenfassung entstand die jährlich stattfindende Document Understandig Conference (DUC). Durchgeführt wird diese von ARDA gesponserte Konferenz von der NIST-Agentur, um weitere Fortschritte auf dem Gebiet der Zusammenfassung zu erzielen und um den Forschern die Teilnahme an Experimenten im großen Rahmen zu ermöglichen.

4.2.3.1.1. DUC 2001

Bei der DUC-2001 im September 2001 wurden generische, single- und multidokument Zusammenfassungen von wissenschaftlichen Veröffentlichungen mit intrinsischen Methoden evaluiert. Die Länge der Zusammenfassungen war dabei festgelegt auf 50, 100, 200 und 400 Worte.

4.2.3.1.2. DUC 2002

Im Juli 2002 fand in Philadelphia als ein Teil des ACL-02 Automatic Summarization Workshops die DUC-2002 statt. Die Aufgaben für die Zusammenfassungssysteme waren die automatische Zusammenfassung eines Zeitungsartikels zu einer 100 Worte langen Zusammenfassung, die automatische Generierung einer Zusammenfassung mit vorgegebener Länge aus mehreren Zeitungsartikeln sowie ein oder zwei Pilotprojekte für extrinsische Evaluation. Die Evaluation von Abstrakten wurde manuell und die von Extrakten automatisch durchgeführt.

4.2.3.1.3. DUC 2003

Bei der DUC-2003 mussten die Zusammenfassungssysteme die folgenden vier Aufgaben lösen:

- Sehr kurze Zusammenfassungen: Die Zusammenfassung durften nur bis zu 10 Worte enthalten. Die Evaluation erfolgte intrinsisch mit SEE
- Kurze Zusammenfassungen: Die Länge der Zusammenfassung betrug circa 100 Worte. Die Evaluation erfolgte ausschließlich intrinsisch.
- Kurze Zusammenfassungen auf einen Gesichtspunkt konzentriert: Die 100-Worte-lange Zusammenfassung wurde auf Basis eines Dokumentenclusters und einer Gesichtspunktbeschreibung, welche in einem String die wichtigen Aspekte des Clusters enthielt, die in der Zusammenfassung vorkommen sollten, generiert. Auch hier erfolgte eine intrinsische Evaluierung.
- Kurze, eine Frage beantwortende Zusammenfassungen: Das Zusammenfassungssystem erhielt ein Dokumentencluster, eine Frage und eine Menge von Sätzen, die für die Frage relevant waren, und erzeugte daraus eine circa 100 Worte umfassende Zusammenfassung, die die Frage beantwortete. NIST evaluierte diese Zusammenfassungen intrinsisch und ordnete sie zusätzlich abhängig von ihrer Antwortfähigkeit in unterschiedliche Kategorien.

Für die intrinsische Evaluation wurde nach einem festgelegten Evaluationsprotokoll vorgegangen (siehe Anhang 1).

4.2.3.1.4. DUC 2004

Im Mai 2004 fand in Boston die DUC-2004 statt. Es mussten 5 Zusammenfassungsaufgaben für Zeitungsartikel der TDT- und der TREC-Sammlung von den Systemen gelöst werden. Die

ersten beiden waren die gleichen wie bei der DUC-2003. Die dritte und vierte Aufgabe waren neu und hatten als Input für die Zusammenfassungssysteme automatisch produzierte Übersetzungen vom Arabischen ins Englische. Bei der dritten Aufgabe sollten sehr kurze Zusammenfassungen von einzelnen Dokumenten entstehen. Zu lange Zusammenfassungen wurden abgeschnitten, aber auch zu kurze Zusammenfassungen wurden nicht belohnt. Die Evaluierung erfolgte unter Nutzung des ROUGE N-gram Matchings. Bei der vierten Aufgabe sollten kurze Zusammenfassungen von mehreren Dokumenten entstehen unter der Berücksichtigung von bestimmten Ereignissen. Die Behandlung zu langer Zusammenfassungen sowie die Evaluation erfolgten auf die gleiche Weise wie bei der dritten Aufgabe. Die fünfte Aufgabe bestand darin kurze, Fragen beantwortende Zusammenfassungen zu generieren. Die Frage, die es zu beantworten galt, war „Wer ist X?“, wobei X der Name einer Person oder Gruppe ist. Auch hier erfolgte eine intrinsische Evaluation.

Unterschiede zur DUC-2003 waren die Angabe der Länge der Zusammenfassungen in Byte und nicht mehr in Worten und die Evaluation der ersten bis vierten Aufgabe wurde mit Hilfe von Recall-Oriented Understudy for Gisting Evaluation (ROUGE) durchgeführt.

4.2.3.1.5. DUCs Evaluationsprozedur

Folgende Prozedur wird für die manuelle intrinsische Evaluation bei der DUC genutzt:

1) Die Testperson liest die gesamte Eingabemenge und erstellt dafür eine 100 Worte umfassende Zusammenfassung. Diese ist das Modell.
2) Das Modell wird automatisch in Inhaltseinheiten unterteilt.
3) Die zu evaluierende Zusammenfassung (Peer genannt) wird automatisch in Sätze zerlegt.
4) Unter Beachtung folgender Anweisungen evaluiert ein menschlicher Beurteiler die Zusammenfassung: Für jede Inhaltseinheit:
    a) Finde alle Peereinheiten, die wenigstens einige Fakten des Modells enthalten, und markiere diese.
    b) Denke an alle markierten Peereinheiten und beantworte die Frage: Alle markierten Peereinheiten zusammen, entsprechen zu k% der Meinung, die durch die gerade betrachtete Modelleinheit ausgedrückt wird. Das k kann 0, 20, 40, 60, 80 oder 100 sein.

Der Wert einer Peerzusammenfassung ergibt sich aus dem Durchschnitt aller Werte für die einzelnen Inhaltseinheiten des Modells.

Die Auswahl der Einheiten wird durch den Gebrauch des Summary Evaluation Environment (SEE) erleichtert.

4.2.3.1.6. Nachteile der DUC-Evaluationsprozedur und Verbesserungen

Obwohl allgemein bekannt ist, dass der Vergleich mit nur einer Referenzzusammenfassung keine zuverlässigen Ergebnisse liefert, wird bei der DUC-Evaluation darauf zurückgegriffen. Somit ist der Grad an Übereinstimmungen zwischen Modellzusammenfassung und Peer sehr gering, was folglich zu niedrigen Werten bei der Evaluation führt.
Ein weiterer Kritikpunkt ist die Entscheidung der prozentualen Überdeckung der Meinungen der Modellinhaltseinheit und der Inhaltseinheiten der zu evaluierenden Zusammenfassung. Dieser Prozess ist schwierig und nicht sehr zuverlässig.
Um diese Nachteile zu beseitigen, griff man bei der DUC-2004 zusätzlich auf ROUGE zur Evaluation zurück und 2005 soll die von Ani Nenkova und Rebecca Passonneau entwickelte Evaluierungsmethode – die Pyramiden-Evaluation – genutzt werden.

4.2.3.2. TIPSTER Text Summarization Evaluation (SUMMAC)

Auch die TIPSTER Text Summarization Evaluation (SUMMAC) ist eine Initiative der DARPA, an der verschiedene amerikanische Regierungsagenturen sowie weitere Forschungs- und Wirtschaftsinstitutionen beteiligt sind. Das Ziel von TIPSTER SUMMAC war das Vorantreiben der aktuellen Entwicklungen auf den Gebieten der Informationsrückgewinnung, der Informationsextraktion und der Automatischen Textzusammenfassung.
Als 1992 die erste Phase des TIPSTER-Programms begann, lag das Hauptaugenmerk der Forschung auf der Informationsrückgewinnung und der Informationsextraktion. In dieser Phase unterstützte das TIPSTER-Programm die Message Understandig Conference und die Text Retrieval Conference bei der Evaluierung von Extraktions- und Retrievalsystemen.
In der zweiten Phase des TIPSTER-Programms lag der Fokus auf der Entwicklung einer allgemeinen Architektur, die eine Vernetzung von Informationsrückgewinnung und Informationsextraktion ermöglichen sollte.
Erst in der dritten Phase des TIPSTER-Programms, welche 1996 startete, waren die Volltextzusammenfassung und deren Evaluation zur Hauptaufgabe geworden. Natürlich wurden auch die Forschungen auf dem Gebiet der Extraktion und des Retrievals fortgesetzt. Somit endete im Mai 1998 die erste entwicklerunabhängige Evaluation von Automatischen Textzusammenfassungssystemen, die in so großem Umfang stattfand. Bestandteile dieser

Evaluation waren zwei extrinsischen Evaluationsaufgaben, eine Adhoc-Aufgabe und eine Kategorisierungsaufgabe, sowie eine intrinsische Frage-Antwort-Aufgabe. An der Evaluation haben 16 verschiedene Systeme teilgenommen:

- Carnegie Group Inc. und Carnegie-Mellon University
- Cornell University und SabIR Research Inc.
- GE Research and Development
- New Mexico State University
- University of Pennsylvania
- University of Southern California – Information Sciences Institute
- Lexis-Nexis
- University of Surrey
- IBM Thomas J. Watson Research
- Text Wise LLC
- SRA International
- British Telecommunications
- Intelligent Algorithms
- University of Massachusetts - Center for Intelligent Information Retrieval
- Center for Information Research
- National Taiwan University

Das Ziel dieser letzten Phase des TIPSTER-Programms war einerseits die Bewertung der unterschiedlichen Textzusammenfassungssysteme. Andererseits sollten aber auch Erkenntnisse zu allgemeinen Fragen der Erstellung und Evaluierung von Zusammenfassungssystemen gewonnen werden. Ein weiteres Ziel war es, die Forschungsarbeiten in eine Richtung zu lenken, in der mehr auf die Voraussetzungen und Anforderungen von echten Aufgaben eingegangen wird.

4.2.3.2.1. Kategorisierungsaufgabe

Bei dieser Evaluierungsaufgabe stand die Frage im Mittelpunkt, ob eine schnelle und korrekte Kategorisierung eines Dokuments auch dann möglich ist, wenn von dem besagten Dokument nur eine allgemeine, indikative Zusammenfassung vorliegt. Die Bewerter mussten das Dokument, welches sowohl eine Zusammenfassung als auch ein Volltext sein konnte, in eine von fünf möglichen Kategorien einordnen oder entscheiden, dass es in keine Kategorie passt.

4.2.3.2.2. Ad-hoc-Aufgabe

Bei der Ad-hoc-Aufgabe handelte es sich um eine Retrievalaufgabe, welche indikative, auf den Benutzer ausgerichtete Zusammenfassungen von Zeitungsartikeln evaluieren sollte. Die Frage, die dabei im Raum stand, war, ob die Anfrage-relevante Zusammenfassung wirksam die tatsächliche Relevanz des Dokumentes widerspiegelt. Den Bewertern wurden auch hier Zusammenfassungen und Volltexte zur Einschätzung vorgelegt. Allerdings musste dann für das Dokument und ein vorgegebenes Thema entschieden werden, wie relevant das Dokument für dieses Thema ist.

Task	Intent	Focus	Coverage	Evaluation Decision	Quantitative measures
Categorization	Indicative	Generic	Single document	appropriate category	time accuracy
Adhoc	Indicative	User-directed	Single document	relevant to topic	time accuracy

Abbildung 38: Übersicht über Categorization- und Adhoc-Aufgabe [Firmin 1997]

4.2.3.2.3. Frage-Antwort-Aufgabe

Die letzte Evaluierungsaufgabe war eine intrinsische Frage-Antwort-Aufgabe, welche so angelegt war, dass sie als Unterstützung für einen Analytiker dienen sollte, der einen Bericht schreibt. Für eine themen-bezogene, informative Zusammenfassung eines Dokumentes sollte bei dieser Evaluation der Informationsgehalt bestimmt werden, indem eine Reihe von themenbezogenen Fragen, deren Antworten im Originaldokument enthalten sind, mit Hilfe der Zusammenfassung beantwortet werden.

4.2.3.2.4. Durchführung

Für die Kategorisierungsaufgabe wurden 10 Themen und je 100 Dokumente pro Thema aus der TREC-Sammlung ausgewählt. Dabei ließen sich die 10 Themen in 2 sich ausschließende Bereiche teilen. Die Themen eines Bereiches waren allerdings ähnlich. Deshalb wurden Dokumente, die zu mehreren Themen passten, ausgeschlossen.

Für die Ad-hoc-Aufgabe wurden 20 Themen und je 50 Dokumente pro Thema ausgesucht.

Für beide Aufgaben wurden Zusammenfassungen mit fester Länge (10 % des Originals) und mit variabler Länge verwendet.

Da bei dieser Evaluation getestet werden sollte, ob mit Hilfe von Zusammenfassungen genau so präzise Entscheidungen getroffen werden können, allerdings in kürzerer Zeit, wurden verschiedene Tests gemacht.

Test der Zusammenfassungskonditionen: Dieser Test sollte aufdecken, ob die Performanz (Zeit und Präzision) des Bewerters von verschiedenen Bedingungen beeinflusst wird: Volltext, Zusammenfassung mit vorgegebener Länge, Zusammenfassung mit variabler Länge, Basiszusammenfassung.

Test der teilnehmenden Technologien: Bei diesem Test wurde die Performanz der verschiedenen teilnehmenden Zusammenfassungssysteme verglichen.

Test der Übereinstimmung: Dieser Test sollte zeigen, wie groß die Übereinstimmung zwischen den Entscheidungen der einzelnen Bewerter ist. Dafür wurden den Personen nur Volltextdokumente gezeigt.

Für die Bewertung der Performanz mussten die Zeit, die für die Entscheidung benötigt wurde, und das Ergebnis der Entscheidung betrachtet werden.

Die Zeitmessung erfolgte über die Logdateien. Die Messung der Präzision wurde mit Hilfe von Precision, Recall und F-Wert durchgeführt.

$$Precision = TP/(TP + FP)$$

$$Recall = TP/(TP + FN)$$

$$Fscore = 2 * Precision * Recall/(Precision + Recall)$$

Abbildung 39: Berechnungsvorschriften für Precision, Recall und F-Wert [Mani, Klein, House, Hirschmann, Firmin & Sundheim 2002, S.54]

Ground truth	Subject's judgment	
	Relevant	**Irrelevant**
Relevant is True	TP	FN
Irrelevant is True	FP	TN

Abbildung 40: Tabelle der Möglichkeiten für die Ad-hoc-Aufgabe (TP – richtig positive, FP – falsch positiv, TN – richtig negativ, FN – falsch negativ) [Mani, Klein, House, Hirschmann, Firmin & Sundheim 2002, S.52]

Ground truth	Subject's judgment		
	X	Y	None
X is True	TP	FN	FN
None is True	FP	FP	TN

Abbildung 41: Tabelle der Möglichkeiten für die Kategorisierungsaufgabe (TP – richtig positive, FP – falsch positiv, TN – richtig negativ, FN – falsch negativ; X und Y steht für zwei Kategorien und None für „keine Kategorie ist richtig“) [Mani, Klein, House, Hirschmann, Firmin & Sundheim 2002, S.52]

Für die Frage-Antwort-Aufgabe wurden von den 20 Themen der Ad-hoc-Aufgabe drei ausgewählt. Aus dem Textkorpus der Ad-hoc-Aufgabe wurden anschließend für jedes Thema 30 relevante Dokumente bestimmt. Für jedes Thema wählte je ein Bewerter Fragen aus, markierte die Antworten dazu im Text und bewertete die einzelnen Zusammenfassungen (siehe Anhang 2). Fehlten Teile einer Antwort in der Zusammenfassung konnten die Bewerter diese entweder als „teilweise richtig“ oder als „fehlend“ bewerten. Zur Bewertung der Präzision kamen drei andere Metriken zum Einsatz (vgl. [Mani, Klein, House, Hirschmann, Firmin & Sundheim 2002]):

- ARS (Answer Recall Strict)
  - ARS = n1 / n3
- ARL (Answer Recall Lenient)
  - ARL = (n1 + (0.5 * n2))/n3
- ARA (Answer Recall Average)
  - ARA = (ARS + ARL) / 2

n1 = Anzahl korrekter Antworten in der Zf.
n2 = Anzahl teilweise korrekter Antworten in der Zf.
n3 = Anzahl beantworteter Fragen im Original

Dabei ist die ARL-Metrik etwas nachsichtiger als die ARS-Metrik, da hier auch teilweise richtige Antworten gewertet werden.

4.2.3.2.5. Ergebnisse

Die Ergebnisse zeigten, dass es kaum möglich ist, umfassende Angaben über die Performanz von Zusammenfassungssystemen zu machen. Schuld daran waren laut Hovy die Tatsachen, dass Zusammenfassungssysteme relativ neu waren, dass es so viele verschiedene Arten von Zusammenfassungen gab und vor allem auch dass es so viele Möglichkeiten der Performanzmessung gab.

Die Ergebnisse der einzelnen Tests sahen wie folgend aus.

Ergebnis des Tests der Zusammenfassungskonditionen:

Als Ergebnis der Ad-hoc-Aufgabe zeigte sich, dass durch die Zusammenfassungen mit variabler Länge (bis zu einer Kompression von 17 %) eine Halbierung der Zeit, die für die Entscheidung benötigt wird, mit einer nur geringen Präzisionseinbuße von 4 % erreicht wurde. Bei der Kategorisierungsaufgabe konnte eine Beschleunigung der Entscheidung von erstmals 43.11 Sekunden auf 25.48 Sekunden erreicht werden, was einem Zeitgewinn von 40% entspricht. Unterschiede in der gemessenen Präzision waren nicht signifikant. Die Werte der Basiszusammenfassung, welche aus den ersten 10 % des Originals bestand, waren kaum zu erreichen, da bei den eingesetzten Zeitungsartikeln die wichtigsten Fakten stets am Anfang standen.

Ergebnis des Tests der teilnehmenden Technologien:

Bei der vergleichenden Untersuchung der unterschiedlichen Zusammenfassungssysteme waren die Ergebnisse der Ad-Hoc-Aufgabe sehr nah beieinander. In Abbildung 42 kann man sehen, dass die Präzision sowohl bei Zusammenfassungen mit variabler Länge als auch bei denen mit fest vorgegebener Länge ähnlich groß ist. Signifikante Unterschiede gibt es nur bei Zusammenfassungen mit variabler Länge für die berechneten F-Werte. Die Gruppe CGI/CMU, Cornell/SabIR unterscheidet sich wesentlich von ISI und ist signifikant präziser.

System	S2			S1		
	P	R	F-score	P	R	F-score
CGI/CMU	.82	.66	.72	.76	.52	.60
Cornell/SabIR	.78	.67	.70	.79	.47	.56
GE	.78	.60	.67	.77	.45	.55
LN	.78	.58	.65	.81	.45	.55
Penn	.81	.57	.65	.76	.45	.53
UMass	.80	.54	.63	.81	.47	.56
NMSU	.80	.54	.63	.80	.40	.52
TextWise	.81	.51	.61	.79	.41	.52
SRA	.82	.49	.60	.79	.37	.48
NTU	.80	.49	.59	.82	.34	.46
ISI	.80	.46	.56	.82	.36	.47

Abbildung 42: Genauigkeit bei der Ad-Hoc-Aufgabe der verschiedenen Zusammenfassungssysteme (S1 – feste Länge; S2 – variable Länge)

Ergebnis des Tests der Übereinstimmung:

Die Übereinstimmungswerte waren bei der Ad-Hoc-Aufgabe (16,6 %) und der Kategorisierungsaufgabe (19,5 %) sehr gering und auch niedriger als erwartet (siehe

Abbildung 43). In ähnlichen Experimenten zuvor, die von Harman und Voorhees 1996 im Rahmen der TREC durchgeführt wurden, konnten höhere Übereinstimmungswerte erreicht werden. Zwei Gründe dafür waren nach Meinung der Forscher von TIPSTER SUMMAC die geringe Erfahrung der ausgewählten Bewerter in solchen Aufgaben und die Auswahl der Texte.

Task	Pairwise	3-way	All 7	All 14
Adhoc	69.1	53.7	NA	16.6
Categorization	56.4	50.6	19.5	NA
Adhoc Dry-Run	72.7	59.1	NA	NA
TREC	88.0	71.7	NA	NA

Abbildung 43: Prozentanteil an übereinstimmenden Entscheidungen (gezeigt wurde der Volltext) [Mani, Klein, House, Hirschmann, Firmin & Sundheim 1999, S. 83]

Ergebnis der Frage-Antwort-Aufgabe:

Die höchsten ARA-Werte konnten für die geringste Kompression (35 – 40 %) erreicht werden. Dabei erhielten die Systeme die besten ARA-Werte, welche bei der Ad-Hoc-Aufgabe in Gruppe I der Genauigkeit waren.

	257			258			271		
Party	ARS	ARL	ARA	ARS	ARL	ARA	ARS	ARL	ARA
CGI/CMU	69	77	73	63	71	67	66	78	72
Cornell/SabIR	61	68	65	64	70	67	60	73	66
GE	49	60	55	58	67	63	56	70	63
NMSU	52	63	58	62	69	65	48	63	55
SRA	36	46	41	61	69	65	35	45	40
ISI	30	42	36	55	59	57	22	34	28
TextWise	27	34	30	32	40	36	29	41	35
Penn	17	24	21	48	55	51	24	34	29

Abbildung 44: Answer Recall – Werte für die unterschiedlichen System bei 3 verschiedenen Themen [Mani, Klein, House, Hirschmann, Firmin & Sundheim 1998, S. 37]

Die Frage-Antwort-Aufgabe zeigte im SUMMAC-Projekt ein viel versprechendes Potential für die Evaluierung von informativen Zusammenfassungen.

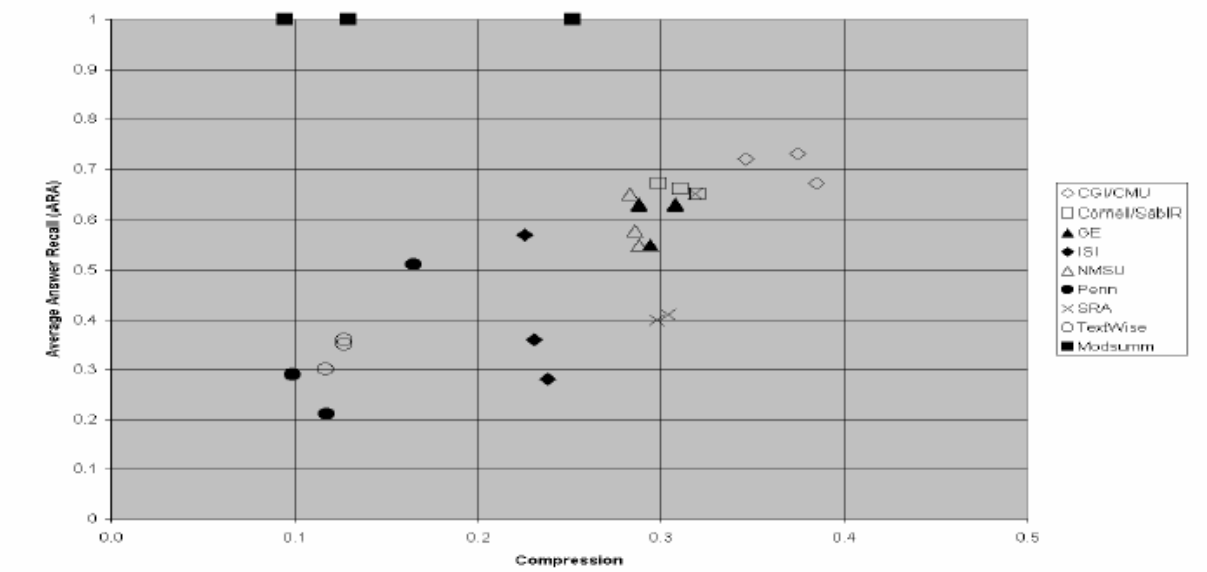


Abbildung 45: Zusammenhang zwischen ARA und Kompression [Mani, Klein, House, Hirschmann, Firmin & Sundheim 1998, S. 38]

4.3. Evaluation des implementierten Zusammenfassungssystems

Mit Hilfe einer Evaluierung wollten wir die verschiedenen implementierten Zusammenfassungsalgorithmen, welche auf unterschiedlichen Extraktionsverfahren basieren, vergleichen. Hierbei sollten auch unterschiedliche Parametrisierungen untersucht werden. Des Weiteren sollte außerdem ein Vergleich mit dem kommerziellen Zusammenfasser von Microsoft ® Word 2000 erfolgen.

4.3.1. Auswahl der Evaluierungsmethode

Um nicht nur die Qualität sondern auch die Nützlichkeit der erzeugten Zusammenfassungen bewerten zu können, wären sowohl ein intrinsisches als auch ein extrinsisches Evaluationsverfahren von Nöten. Dies würde aber zu einer viel zu umfangreichen Evaluierung mit zu hohem Zeitaufwand sowohl für die Evaluierungsteilnehmer als auch für die Auswertung führen.

Daher entschieden wir uns für eine rein intrinsische Methode, welche in ähnlicher Form auch von den meisten anderen Forschern auf diesem Gebiet genutzt wurde.

In diesem intrinsischen Evaluierungsschritt sollten die Testpersonen selbst Zusammenfassungen erstellen, indem sie in verschiedenen Texten die für sie wichtigsten Sätze markieren. Dabei sollte es keine strikte Vorgabe der Anzahl der Sätze geben, um die Testpersonen nicht zu stark einzuschränken oder zu beeinflussen. Es wurde vorgegeben, dass zwischen 5 und 15 Sätze markiert werden sollen.

Anhand dieser Zusammenfassungen sollten die wichtigsten Sätze jedes Textes ermittelt werden und über Precision und Recall mit den automatisch generierten Zusammenfassungen verglichen werden.

4.3.2. Auswahl der Texte für die Evaluierung

4.3.2.1. Auswahl des Texttyps

Zur Auswahl des Texttyps standen verschiedene Kategorien, wie Zeitungsartikel, Lehrbuchtexte oder wissenschaftliche Arbeiten. Andere Texttypen wie zum Beispiel Bedienungsanleitung, Gedicht oder Märchen schieden von vornherein aus, weil sie sich nicht zum Zusammenfassen eignen, da hier das Weglassen von Sätzen den gesamten Sinn des Textes zerstören würde.
Unsere Entscheidung fiel auf Zeitungsartikel, welche sowohl in gedruckten Zeitungen als auch in Webzeitungen zu finden sein können, da bei diesen Texten keine spezifische Vorbildung benötigt wird und da diese Texte meist zwischen 1 und 1,5 Seiten lang sind. Das heißt, sie sind nicht zu lang und nicht zu kurz. So können einerseits die Evaluierungsteilnehmer sie in kurzer Zeit durchlesen und ihre eigene Zusammenfassung erstellen und andererseits macht es auch noch Sinn die Texte zusammenzufassen.

4.3.2.2. Auswahl der Textthemen

Wir entschieden uns für drei verschiedene Zeitungsartikel zu drei sehr unterschiedlichen Themen. Wir legten uns auf einen Text aus dem Bereich der Politik fest, bei welchem wir erwarteten, dass das Interesse der Teilnehmer an diesem Thema recht gering sein wird. Der zweite Text behandelte das relativ aktuelle Thema der Vogelgrippe in der Türkei. Bei diesem Thema rechneten wir mit größerem Interesse der Evaluierungsteilnehmer, da bei einer möglichen Vogelgrippepandemie in Deutschland jeder davon betroffen wäre und dadurch eine rege Beschäftigung mit diesem Thema stattfand. Als drittes Thema wählten wir den Gerichtsprozess über den Mord an der sechsjährigen Ayla im April 2005. Bei diesem Thema erwarteten wir ein sehr großes Interesse der Evaluationsbeteiligten, da dies ein sehr emotionales Thema ist und da es um ein Verbrechen aus der unmittelbaren Umgebung geht.
Durch die unterschiedlichen Themenbereiche und das damit verbundene unterschiedliche emotionale Interesse wollten wir herausfinden, ob diese Faktoren die Teilnehmer bei der Erstellung ihrer Zusammenfassungen beeinflussen.

4.3.2.3. Auswahl des Textformats

Obwohl unser Zusammenfassungsprogramm sowohl Texte im RTF-Format sowie im TXT-Format als auch im HTML-Format verarbeiten kann und gerade unser Extraktionsalgorithmus speziell auf die Besonderheiten von HTML-Texten eingeht, wählten wir für die Evaluierung Texte im RTF-Format aus. Dies hatte den einfachen Grund, dass die Evaluierung in einem Hörsaal und nicht in einem Computerpool durchgeführt werden musste und dadurch Eigenheiten der HTML-Text-Zusammenfassung wie zum Beispiel die Berücksichtigung von Linkinhalten bei der Satzauswahl nicht beachtet werden konnten.

4.3.3. Durchführung der Evaluierung

Am ersten Teil der Evaluation am 24.01.06 um 10 Uhr nahmen 28 Personen teil und am zweiten Teil am 31.01.06 um 18 Uhr noch einmal 14 Personen.
Jeder Teilnehmer erhielt die drei unterschiedlichen Zeitungsartikel und eine Anleitung, die besagte, dass in allen drei Texten die 5 bis 10 bzw. 5 bis 15 wichtigsten Sätze zu markieren sind (siehe Anhang 3 – 5). Dafür erhielten die Testpersonen 30 Minuten Zeit.

4.3.4. Vergleichszusammenfassungen

Die Zusammenfassungen, die mit der Referenzzusammenfassung verglichen und bewertet werden sollten, wurden zum großen Teil durch unser Programm zur automatischen Erstellung von Zusammenfassungen erstellt. Eine Zusammenfassung wurde mit der Zusammenfassungsfunktion des kommerziellen Textverarbeitungsprogramms Word® generiert.
Bei allen Zusammenfassungen wählten wir für die Größe der Zusammenfassung 20 % des Originaltextes.
Bei den Algorithmen, die über unterschiedliche Parameter gesteuert werden können, wählten wir verschiedene Parametrisierungen. Außerdem wurden jeweils Zusammenfassungen mit und ohne Angabe von Suchworten generiert.

Luhn-Algorithmus	- mit und ohne Suchworte - mit Grenzwert oder Wortliste
Edmundson-Algorithmus	- ohne Suchworte - unterschiedlich Cue-, Location-, Key- und Title-Gewichte

Satzgewicht-Algorithmus	- mit und ohne Suchworte - unterschiedliche Gewichte für Satzposition, Satzlänge, Abschnitt, enthaltene Suchworte, wichtige bzw. unwichtige Phrasen und Fragen
Genetischer Algorithmus	- mit und ohne Suchworte - unterschiedliche Gewichte für Satzposition, Satzlänge, Abschnitt, enthaltene Suchworte, wichtige bzw. unwichtige Phrasen und Fragen
Eigener Algorithmus	- mit und ohne Suchworte - unterschiedliche Gewichte für Positionswert, Überschrifteninhaltswert, Suchwortwert und Eigennamenwert

Der Euler-Algorithmus konnte nicht zum Einsatz kommen, da für die drei unterschiedlichen Textthemen im Vorfeld Stichwortlisten für diese drei Domänen recht aufwendig hätten generiert werden müssen.

4.3.5. Ergebnisse der Evaluierung

4.3.5.1. Auswertung der Nutzerbefragung

Nachdem die Teilnehmer die wichtigsten Sätze der drei verschiedenen Texte markiert hatten, begannen wir auszuzählen, wie viele Sätze durchschnittlich gekennzeichnet wurden und welche Sätze am häufigsten selektiert wurden (siehe Tabelle 2 – 4).
So zeigte sich, dass im ersten und dritten Text, welche ungefähr die gleiche Länge hatten und bei denen 5 bis 10 Sätze markiert werden sollten, fast die gleiche Anzahl an durchschnittlich markierten Sätzen ermittelt wurde.

Durchschnittlich Anzahl selektierter Sätze im ersten Text: $n_{Durchschnitt} = \frac{\sum_{1}^{42} n_i}{42} = \frac{275}{42} = \underline{\underline{6{,}55}}$

Durchschnittlich Anzahl selektierter Sätze im dritten Text: $n_{Durchschnitt} = \frac{\sum_{1}^{42} n_i}{42} = \frac{267}{42} = \underline{\underline{6{,}36}}$

Das zeigt, dass das von uns als unterschiedlich eingestufte Interesse der Teilnehmer an den Themen dieser beiden Texte keine Auswirkungen auf die Anzahl der als wichtig deklarierten Sätze hatte.
Die durchschnittliche Anzahl als wichtig gekennzeichneter Sätze im zweiten Text, welcher circa 50% länger war als die beiden anderen und bei welchem zwischen 5 und 15 Sätzen markiert werden sollten, lag auch ungefähr 50% über den anderen Durchschnittswerten.

Durchschnittliche Anzahl selektierter Sätze im zweiten Text: $n_{Durchschnitt} = \frac{\sum_{1}^{42} n_i}{42} = \frac{427}{42} = \underline{\underline{10{,}17}}$

In den drei folgenden Diagrammen ist die Häufigkeit der Markierung der einzelnen Sätze dargestellt.

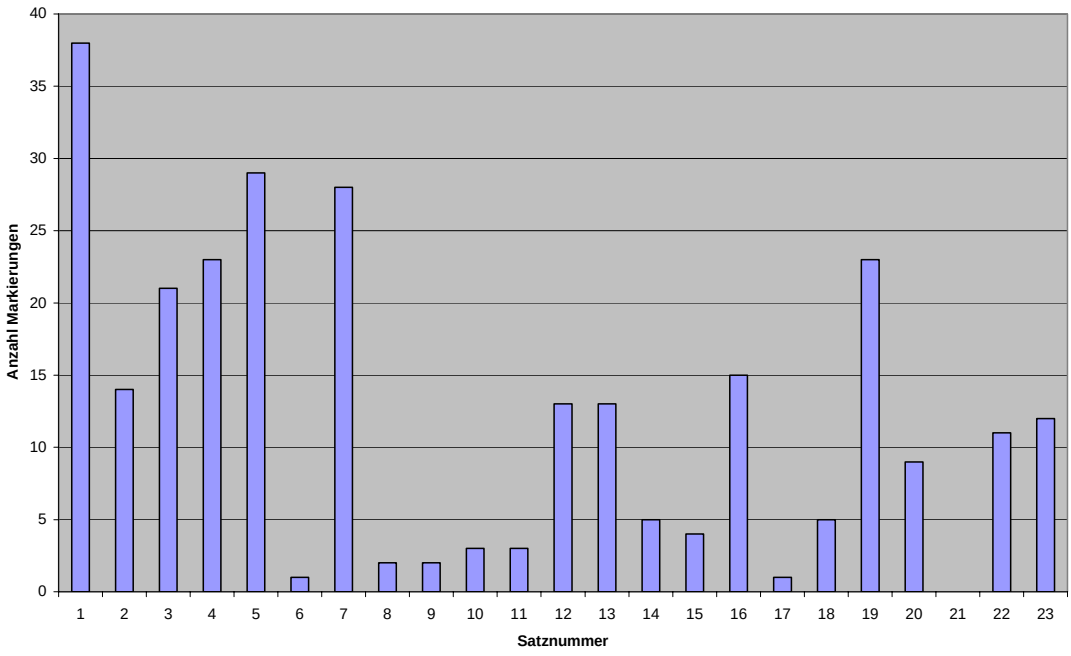


Abbildung 46: Übersicht über die Häufigkeit der Markierung der einzelnen Sätze im ersten Text

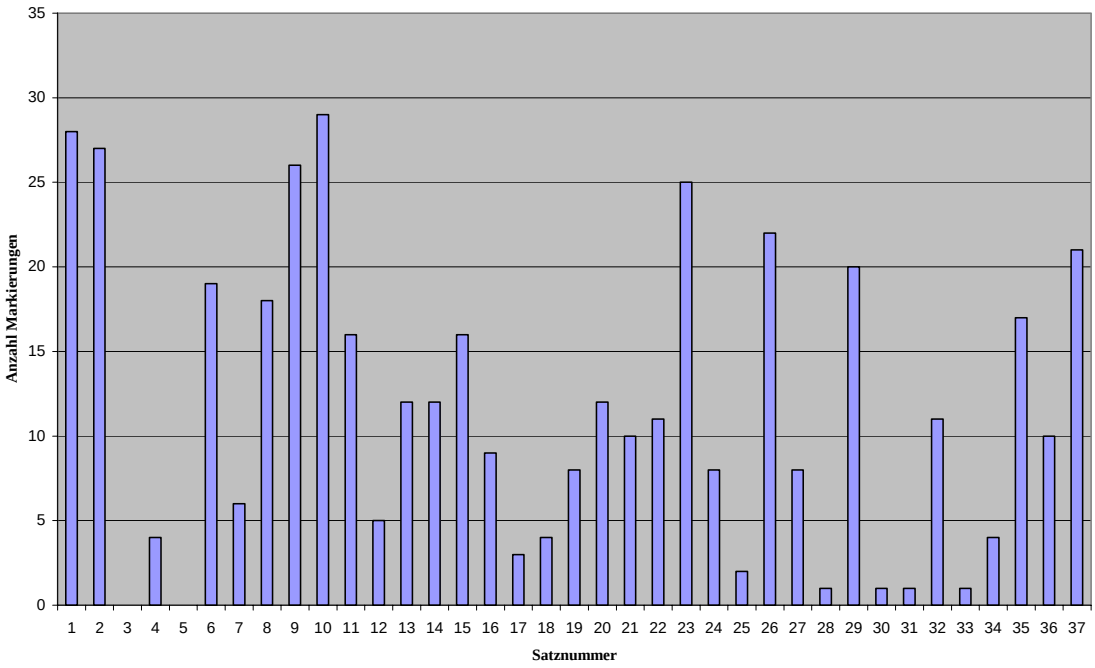


Abbildung 47: Übersicht über die Häufigkeit der Markierung der einzelnen Sätze im zweiten Text

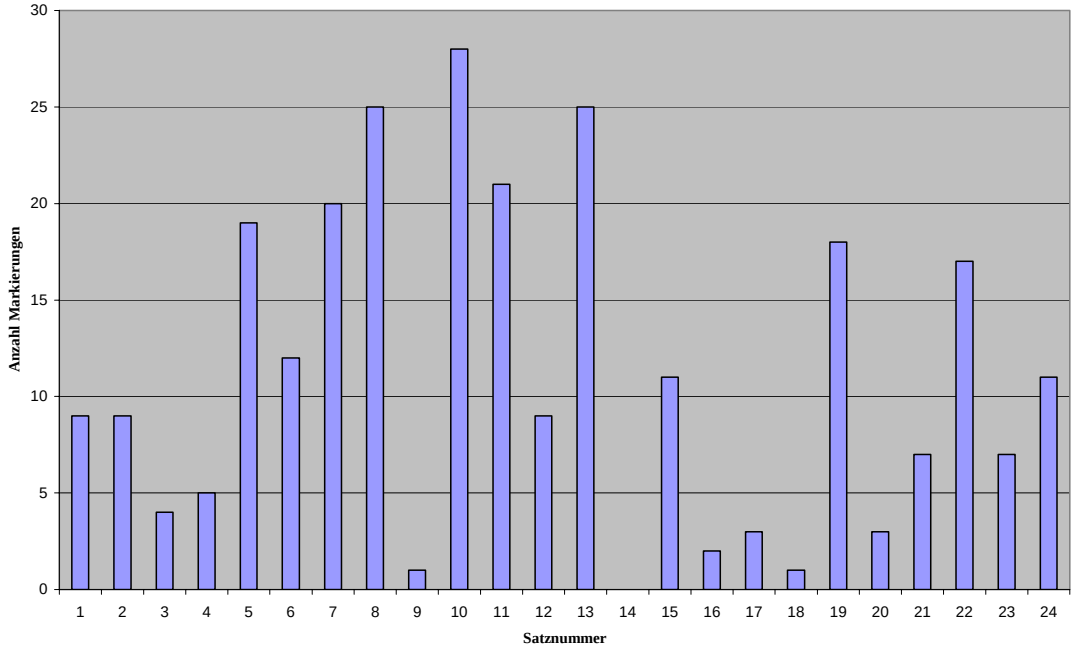


Abbildung 48: Übersicht über die Häufigkeit der Markierung der einzelnen Sätze im dritten Text

Aus diesen Angaben bestimmten wir mit Hilfe des Originaltextes die Beispielzusammenfassungen. Das heißt, wenn im ersten Text durchschnittlich rund 7 Sätze markiert wurden, wählten wir die 7 Sätze mit der höchsten Anzahl an Markierungen als Beispielzusammenfassung aus.

SatzNr	Anz. Markierung
1	38
2	14
3	21
4	23
5	29
6	1
7	28
8	2
9	2
10	3
11	3
12	13
13	13
14	5
15	4
16	15
17	1
18	5
19	23
20	9
21	0
22	11
23	12

Tabelle 1: Hervorhebung der wichtigsten Sätze im ersten Text

Damit ergab sich folgende Referenzzusammenfassung:

> ***Lebenslange Haft für Ayla-Mörder***
>
> *Der Mörder der sechsjährigen Ayla ist zu lebenslanger Haft verurteilt worden. Damit ist eine vorzeitige Freilassung des Täters nach 15 Jahren Haft ausgeschlossen. Der Verurteilte, ein 37 Jahre alter Mann, war wegen Mordes und sexuellen Missbrauchs vorbestraft. Er hatte gestanden, Ayla im Mai vergangenen Jahres auf dem Weg zur Schule entführt, sexuell misshandelt und getötet zu haben.*
>
> ***Sicherheitsverwahrung wurde nicht verhängt***
>
> *Der Forderung nach Sicherheitsverwahrung wurde allerdings nicht entsprochen.*
>
> ***Angeklagter saß bereits wegen Missbrauch im Gefängnis***
>
> *Ayla war im Mai 2005 auf dem Weg zur Schule in ein Auto gezerrt, sexuell missbraucht und ermordet worden. Der 37-jährige Mann ist bereits einschlägig vorbestraft.*

Auf diese Art und Weise erstellten wir auch für die restlichen beiden Texte die Referenzzusammenfassungen (siehe Anhang 6 und 7). Mit diesen Zusammenfassungen wurden die automatisch generierten Zusammenfassungen zur Bewertung ihrer Qualität verglichen.

Teilnehmer Nr / SatzNr	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	Gesamt-anzahl
1	X	X	X	X	X	X	X	X	X		X	X	X	X	X	X		X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X			X	X	X	X	X	X	38
2													X	X			X	X	X		X		X				X		X	X	X		X							X	X	14
3	X		X	X		X	X		X				X	X	X			X	X		X		X	X	X		X			X			X			X					X	X	21
4	X	X	X		X	X		X	X	X	X					X				X	X		X			X	X			X		X			X	X		X	X	X		X	23
5	X	X	X	X	X	X		X	X	X			X	X		X	X		X	X	X		X			X		X	X	X	X			X	X		X	X	X		X	X	29
6		X																																									1
7	X	X		X		X	X	X			X		X	X			X	X	X	X	X	X	X	X	X	X	X	X	X	X		X				X		X	X			28
8			X													X																											2
9			X										X																														2
10						X																						X				X										3
11						X																						X				X										3
12			X	X						X					X	X								X					X	X			X	X			X			X	X		13
13												X			X	X						X				X	X		X			X			X	X		X	X			X	13
14																	X			X													X					X				X	5
15																	X								X							X										X	4
16			X		X		X	X	X						X			X	X					X		X				X	X	X			X				X				15
17																															X												1
18												X																			X				X	X				X			5
19			X	X		X	X	X		X		X	X		X	X	X	X					X		X	X			X				X		X	X		X		X	X	X		23
20			X	X		X		X														X												X				X	X	X			9
21																																												0
22			X			X		X		X												X							X	X			X	X				X		X			11
23	X	X										X			X		X								X			X	X									X	X	X	X		12
Anz:	6	6	11	7	4	10	5	8	5	5	3	5	7	5	7	7	7	6	6	5	6	6	6	6	6	6	6	7	7	9	8	8	7	6	6	4	5	8	9	9	7	8	

Tabelle 2: Selektierte Sätze im Text „Lebenslang für Ayla-Mörder“

Teilnehmer Nr / SatzNr	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	Gesamt-anzahl
1	X	X		X	X	X	X	X	X		X		X	X		X	X		X	X		X	X	X	X	X	X			X	X	X	X						X		X	X	28
2	X	X		X	X	X	X	X	X	X	X		X						X	X	X	X	X	X	X	X	X			X	X	X	X	X				X		X			27
3																																											0
4			X																		X									X							X						4
5																																											0
6			X	X					X	X					X		X	X	X		X				X	X		X		X			X		X	X	X	X		X			19
7									X			X					X	X		X													X										6
8				X								X		X	X	X		X	X							X	X	X	X	X	X	X			X			X	X			X	18
9	X	X	X	X		X		X	X		X		X			X	X	X	X	X	X	X	X	X	X							X	X			X	X	X	X	26
10	X	X	X				X	X	X		X	X		X	X		X	X	X	X	X	X	X				X		X	X	X	X		X			X	X	X	X	X	X	29
11				X	X	X		X			X				X	X	X						X		X	X			X		X		X		X	X							16
12	X										X												X		X											X								5
13			X	X						X	X		X									X	X	X				X		X					X							X	12
14										X		X	X	X								X	X						X	X								X	X		X	X	12
15			X				X		X	X	X					X			X		X	X					X				X	X		X			X			X		X	16
16			X								X	X			X		X														X	X	X						X			9
17																						X																X			X			3
18			X															X							X													X					4
19	X	X				X					X			X								X							X		X							X					8
20	X	X								X	X								X			X			X			X	X		X								X	X		12
21	X			X		X						X						X	X						X			X	X										X			10
22	X					X				X								X	X			X				X	X	X	X										X			11
23	X		X	X		X	X	X			X			X		X	X	X	X			X		X		X	X		X					X	X		X	X	X		X	25
24	X	X																	X					X	X			X							X			X					8
25																			X																			X					2
26	X	X		X				X	X		X		X		X	X		X				X		X			X	X	X	X		X	X				X	X		X	X	22
27								X	X					X								X						X	X						X				X			8
28																																	X										1
29	X	X	X			X	X			X	X		X		X	X	X						X							X	X	X	X	X			X	X			X	20
30										X																																	1
31							X																																				1
32					X			X			X	X	X					X		X		X										X		X			X					11
33																														X													1
34																		X	X									X										X					4
35			X			X		X	X		X	X		X	X	X	X			X			X	X							X				X				X	X			17
36	X	X	X			X									X	X				X			X			X								X									10
37	X	X	X		X	X	X		X					X	X	X			X	X			X	X		X		X	X					X		X	X						21
Anz:	15	11	13	10	5	12	8	10	11	9	16	8	8	9	10	11	10	12	15	9	8	15	11	8	10	10	9	12	12	10	11	10	11	8	6	7	7	13	7	12	8	10	

Tabelle 3: Selektierte Sätze des Textes "Tödliches Spiel mit Hühnerblut "

Teilnehmer Nr / SatzNr	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	Gesamt-anzahl
1					X										X			X											X		X	X	X	X				X				9
2					X											X		X										X	X	X	X	X					X				9
3					X													X												X								X				4
4				X								X	X										X				X															5
5			X			X	X	X		X				X		X		X					X	X		X	X	X	X	X	X	X	X	X								19
6						X		X						X	X		X		X		X		X								X		X			X				X	12
7		X		X		X	X				X		X			X		X	X	X		X				X	X	X	X	X					X		X	X		X			20
8	X	X		X		X	X	X		X	X	X			X	X		X	X	X	X	X	X							X	X	X			X	X		X	X			X	25
9								X																																			1
10		X	X		X					X	X	X	X	X	X	X	X	X	X	X	X	X	X			X	X	X	X				X			X	X	X		X	X	X	28
11	X			X			X	X			X			X	X	X	X	X	X	X	X	X	X							X	X		X		X					X	X		21
12											X										X						X							X	X			X	X	X	X		9
13	X	X		X			X	X			X		X	X	X	X		X	X	X	X						X	X	X		X	X	X		X	X		X	X	X			25
14																																											0
15	X	X				X							X		X	X		X			X									X						X				X			11
16																				X																X							2
17				X													X			X																						3
18												X																															1
19			X	X		X						X	X		X	X	X	X		X	X					X	X			X	X	X						X				X	18
20		X						X																														X			X		3
21	X	X						X		X												X									X								X				7
22				X			X										X	X	X		X	X					X	X	X		X		X		X	X		X	X			X	17
23																X			X	X												X		X			X			X			7
24										X						X		X	X		X						X	X	X				X				X		X				11
Anz.:	5	7	3	8	4	6	6	8	0	5	6	5	6	5	8	11	6	13	9	9	10	6	6	1	0	4	9	6	6	8	8	9	8	6	7	5	6	7	8	5	7	5	

Tabelle 4: Selektierte Sätze im Text "Die SPD hat ein Herz für Familien"

4.3.5.2. Vergleich mit Referenzzusammenfassungen

Um die verschiedenen Zusammenfassungen miteinander vergleichen zu können, benötigten wir ein objektives Maß zur Bewertung. Dafür nutzten wir die Werte Precision und Recall, wie das auch die meisten Forscher auf diesem Gebiet getan haben.
Diese beiden Maße werden im Information Retrieval oder bei Recherchen genutzt, um die Güte der Suchergebnisse beschreiben zu können. Da auch im Bereich der Automatischen Textzusammenfassung eine Art Recherche durchgeführt wird, bei der aus allen Sätzen die relevanten herausgesucht werden, können diese Maße übernommen werden.
Precision beschreibt dabei die Genauigkeit eines Suchergebnisses. Das heißt, sie ist der Anteil wirklich relevanter Sätze von allen für die Zusammenfassung ausgewählten Sätzen.
Recall hingegen beschreibt die Vollständigkeit eines Suchergebnisses. Das heißt, er ist der Anteil für die Zusammenfassung ausgewählter relevanter Sätze an allen relevanten Sätzen des Textes.

$$\mathrm{Re}\,call = \frac{|\{relevanteSätze\} \cap \{ausgewählteSätze\}|}{|\{relevanteSätze\}|}$$

$$\mathrm{Pr}\,ecision = \frac{|\{relevanteSätze\} \cap \{ausgewählteSätze\}|}{|\{ausgewählteSätze\}|}$$

Abbildung 49: Berechnungsformeln für Precision und Recall

4.3.5.2.1. Zusammenfassungen mit dem Luhn –Algorithmus

Mit dem Luhn-Algorithmus und einer Kompression von 20 % haben wir für jeden Text vier Zusammenfassungen generieren lassen. Zwei mit und zwei ohne Suchworte sowie mit den zwei unterschiedlichen Verfahren zur Auswahl relevanter Worte aus der Liste mit allen Worten.
Die Zusammenfassungen des Ayla-Textes unterscheiden sich dabei höchstens in einem Satz untereinander.

Zusammenfassung	Ausgewählte Sätze	Recall	Precision
Mit Grenzwert und Suchworten	1, 5, 14, 15, 17	$\frac{2}{7} = 0{,}286$	$\frac{2}{5} = 0{,}4$
Mit Grenzwert und ohne Suchworte	1, 6, 14, 15, 17	$\frac{1}{7} = 0{,}143$	$\frac{1}{5} = 0{,}2$
Mit Wortliste und Suchworten	1 6 14 15 17	$\frac{1}{7} = 0{,}143$	$\frac{1}{5} = 0{,}2$
Mit Wortliste und ohne Suchworte	1 5 14 15 17	$\frac{2}{7} = 0{,}286$	$\frac{2}{5} = 0{,}4$

Tabelle 5: Zusammenfassungen des ersten Textes mit dem Luhn-Algorithmus und dazugehörige Recall- und Precisionwerte

Auch die Zusammenfassungen des SPD-Klausur-Textes unterscheiden sich untereinander höchstens in einem Satz

Zusammenfassung	Ausgewählte Sätze	Recall	Precision
Mit Grenzwert und Suchworten	1, 10, 17, 18	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
Mit Grenzwert und ohne Suchworte	1, 10, 16, 18	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
Mit Wortliste und Suchworten	1, 10, 17, 18	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
Mit Wortliste und ohne Suchworte	1, 10, 16, 18	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$

Tabelle 6: Zusammenfassungen des dritten Textes mit dem Luhn-Algorithmus und dazugehörige Recall- und Precisionwerte

Die vier Zusammenfassungen des Vogelgrippe-Textes mit dem Luhn-Algorithmus und unterschiedlichen Parametern sind alle identisch.

Ausgewählte Sätze	Recall	Precision
1, 6, 8, 13, 16, 18, 32	$\frac{3}{10} = 0{,}3$	$\frac{3}{7} = 0{,}429$

Tabelle 7: Zusammenfassungen des zweiten Textes mit dem Luhn-Algorithmus und dazugehörige Recall- und Precisionwerte

Wie man an den ausgewählten Sätzen der verschiedenen Zusammenfassungen erkennen kann, bewirken die unterschiedlichen Verfahren zur Auswahl der relevanten Worte nur sehr wenig

bis gar nichts. Auch die Berücksichtigung von Suchworten, welche die Suche an Nutzerinteressen anpassen soll, beeinflusst die Summaries kaum.
Außerdem ist aus den geringen Precision- und Recall-Werten ersichtlich, dass dieser Algorithmus keine guten Zusammenfassungen liefert. Dies zeigt, dass die bloße Berücksichtigung der Worthäufigkeiten keine sinnvoll nutzbaren Ergebnisse liefert. Allerdings sind die Werte bei längeren Texten besser als bei kurzen.

4.3.5.2.2. Zusammenfassungen mit dem Edmundson –Algorithmus

Mit diesem Algorithmus ließen wir für jeden der drei Texte 5 Zusammenfassungen erstellen. Diese fünf haben jeweils unterschiedliche Parametrisierungen (siehe Tabelle 8). Suchworte werden von diesem Algorithmus nicht berücksichtigt.

Zusammenfassung	Cue-Gewicht	Key-Gewicht	Title-Gewicht	Location
1 (CKTL)	1	1	1	1
2 (C)	1	0	0	0
3 (K)	0	1	0	0
4 (T)	0	0	1	0
5 (L)	0	0	0	1

Tabelle 8: Gewichtungen der einzelnen Methoden bei den verschiedenen Zusammenfassungen

Bei den 5 verschiedenen Zusammenfassungen des Ayla-Textes beeinflussten die unterschiedlichen Parametrisierungen die Satzauswahl merklich.

Zusammenfassung	Ausgewählte Sätze	Recall	Precision
CKTL	4, 5, 13, 16, 22	$\frac{3}{7} = 0{,}429$	$\frac{3}{5} = 0{,}6$
C	1, 2, 3, 4, 5	$\frac{4}{7} = 0{,}571$	$\frac{4}{5} = 0{,}8$
K	4, 5, 13, 16, 22	$\frac{3}{7} = 0{,}429$	$\frac{3}{5} = 0{,}6$
T	1, 5, 13, 16, 22	$\frac{3}{7} = 0{,}429$	$\frac{3}{5} = 0{,}6$
L	1, 2, 16, 23	$\frac{2}{7} = 0{,}286$	$\frac{2}{4} = 0{,}5$

Tabelle 9: Zusammenfassungen des ersten Textes mit dem Edmundson-Algorithmus und dazugehörige Recall- und Precisionwerte

Bei den Zusammenfassungen des zweiten Textes, dessen Thema die Vogelgrippe in der Türkei war, führten die unterschiedlichen Gewichtungen auch zu unterschiedlichen Satzextraktionen.

Zusammenfassung	Ausgewählte Sätze	Recall	Precision
CKTL	6, 8, 16, 23, 26, 27, 32	$\frac{4}{10} = 0{,}4$	$\frac{4}{7} = 0{,}571$
C	1, 2, 3, 4, 5, 6, 7	$\frac{3}{10} = 0{,}3$	$\frac{3}{7} = 0{,}429$
K	6, 8, 16, 23, 26, 27, 32	$\frac{4}{10} = 0{,}4$	$\frac{4}{7} = 0{,}571$
T	6, 8, 10, 11, 14, 15, 23	$\frac{4}{10} = 0{,}4$	$\frac{4}{7} = 0{,}571$
L	1, 32, 33, 34, 35, 36, 37	$\frac{2}{10} = 0{,}2$	$\frac{2}{7} = 0{,}286$

Tabelle 10: Zusammenfassungen des zweiten Textes mit dem Edmundson-Algorithmus und dazugehörige Recall- und Precisionwerte

Auch bei dem dritten Text wirkten sich die unterschiedlichen Methodengewichte entscheidend auf die Auswahl der Sätze aus.

Zusammenfassung	Ausgewählte Sätze	Recall	Precision
CKTL	4, 17, 18, 22	$\frac{0}{6} = 0$	$\frac{0}{4} = 0$
C	1, 7, 8, 20	$\frac{2}{6} = 0{,}333$	$\frac{2}{4} = 0{,}5$
K	6, 10, 17, 22	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
T	4, 9, 17, 18	$\frac{0}{6} = 0$	$\frac{0}{4} = 0$
L	1, 22, 23, 24	$\frac{0}{6} = 0$	$\frac{0}{4} = 0$

Tabelle 11: Zusammenfassungen des dritten Textes mit dem Edmundson-Algorithmus und dazugehörige Recall- und Precisionwerte

Wie eindeutig an den geringen Precision- und Recall-Werten zu erkennen ist, hat der Edmundson-Algorithmus beim dritten Text sehr schlechte Zusammenfassungen generiert. Schuld daran könnten aber auch die Textkürze und die geringe Anzahl an ausgewählten und relevanten Sätzen sein.

4.3.5.2.3. Zusammenfassungen mit dem Satzgewicht –Algorithmus

Mit dem Satzgewicht-Algorithmus ließen wir für jeden Text elf Summaries mit unterschiedlicher Gewichtung der einzelnen Werte generieren.

Zusammenfassung	1	2	3	4	5	6	7	8	9	10	11
$Gewicht_{Satzposition}$	1	1	0	1	1	1	1	0	1	0	0
$Gewicht_{Satzlänge}$	1	1	1	0	1	1	1	0	0	1	0
$Gewicht_{Abschnitt}$	1	1	1	1	0	1	1	0	0	0	0
$Gewicht_{enth.Suchworte}$	1	1	1	1	1	0	1	1	0	0	1
$Gewicht_{enth.SuchworteInÜberschr}$	1	1	1	1	1	0	1	1	0	0	0
$Gewicht_{wicht.Phrasen}$	1	1	1	1	1	1	0	0	0	0	0
$Gewicht_{unwicht.Phrasen}$	-1	-1	-1	-1	-1	-1	0	0	0	0	0
$Gewicht_{Fragen}$	-1	-1	-1	-1	-1	-1	-1	0	0	0	0
Textstruktur beachten	nein	ja	nein	nein	nein	nein	nein	nein	nein	nein	nein

Tabelle 12: Komponentengewichte der einzelnen Zusammenfassungen

Mit diesen Gewichtungen wollten wir den Einfluss der einzelnen Komponenten, die zur Satzgewichtberechnung beitragen, aufzeigen.

Bei der Zusammenfassung des ersten Textes wurden die beiden Suchworte „Ayla“ und „Urteil“ in das Zusammenfassungsprogramm eingegeben.

Zusammenfassung	Ausgewählte Sätze	Recall	Precision
1	1, 4, 5, 6, 16	$\frac{4}{7} = 0{,}571$	$\frac{4}{5} = 0{,}8$
2	1, 4, 5, 6, 16	$\frac{4}{7} = 0{,}571$	$\frac{4}{5} = 0{,}8$
3	1, 4, 5, 6, 16	$\frac{4}{7} = 0{,}571$	$\frac{4}{5} = 0{,}8$
4	1, 4, 5, 6, 16	$\frac{4}{7} = 0{,}571$	$\frac{4}{5} = 0{,}8$
5	1, 4, 5, 15a, 16	$\frac{4}{7} = 0{,}571$	$\frac{4}{5} = 0{,}8$
6	1, 2, 5, 16, 17	$\frac{3}{7} = 0{,}429$	$\frac{3}{5} = 0{,}6$
7	1, 4, 5, 6, 16	$\frac{4}{7} = 0{,}571$	$\frac{4}{5} = 0{,}8$
8	1, 2, 4, 5, 6	$\frac{3}{7} = 0{,}429$	$\frac{3}{5} = 0{,}6$
9	1, 7, 15a, 12, 16	$\frac{3}{7} = 0{,}429$	$\frac{3}{5} = 0{,}6$

10	4, 5, 12, 13, 16	$\frac{3}{7} = 0{,}429$	$\frac{3}{5} = 0{,}6$
11	1, 4 ,5, 6, 11	$\frac{3}{7} = 0{,}429$	$\frac{3}{5} = 0{,}6$

Tabelle 13: Zusammenfassungen des ersten Textes mit dem Satzgewicht-Algorithmus und dazugehörige Recall- und Precisionwerte

Aus der obigen Tabelle ist ersichtlich, dass das Weglassen nur einer Komponente meist nur sehr geringe Auswirkungen auf die Qualität der Zusammenfassung hat, da relevante Sätze wahrscheinlich oftmals gleich mehrere Kriterien erfüllen. Beschränkt man sich aber bei der Satzauswahl auf nur ein oder zwei Komponenten (siehe Zusammenfassungen 8 – 11), dann erhält man schlechtere Recall- und Precisionwerte. Im Vergleich mit den beiden vorangegangenen Algorithmen liegen die Recall- und Precisionwerte deutlich über denen der anderen beiden Algorithmen.

Zusammenfassung	Ausgewählte Sätze	Recall	Precision
1	1, 2, 6 ,8, 11, 32, 33	$\frac{4}{10} = 0{,}4$	$\frac{4}{7} = 0{,}571$
2	1, 2, 6 ,8, 11, 32, 33	$\frac{4}{10} = 0{,}4$	$\frac{4}{7} = 0{,}571$
3	1, 2, 6, 7, 32, 33, 35	$\frac{3}{10} = 0{,}3$	$\frac{3}{7} = 0{,}429$
4	1, 2, 3, 6, 8, 11, 32	$\frac{4}{10} = 0{,}4$	$\frac{4}{7} = 0{,}571$
5	1, 6, 8, 11, 16, 26, 32	$\frac{4}{10} = 0{,}4$	$\frac{4}{7} = 0{,}571$
6	1, 8, 16, 32, 33, 34, 35	$\frac{2}{10} = 0{,}2$	$\frac{2}{7} = 0{,}286$
7	1, 2, 6, 8, 11, 32, 35	$\frac{4}{10} = 0{,}4$	$\frac{4}{7} = 0{,}571$
8	1, 2, 3, 4, 5, 6, 7	$\frac{3}{10} = 0{,}3$	$\frac{3}{7} = 0{,}429$
9	1, 3, 8, 11, 16, 21, 26	$\frac{3}{10} = 0{,}3$	$\frac{3}{7} = 0{,}429$
10	6, 16, 23, 26, 32, 33, 35	$\frac{3}{10} = 0{,}3$	$\frac{3}{7} = 0{,}429$
11	1, 2, 6, 8, 11, 16, 19	$\frac{4}{10} = 0{,}4$	$\frac{4}{7} = 0{,}571$

Tabelle 14: Zusammenfassungen des zweiten Textes mit dem Satzgewicht-Algorithmus und dazugehörige Recall- und Precisionwerte

Bei der Zusammenfassung des zweiten Textes ist sehr gut erkennbar, dass die eingegebenen Suchworte sehr stark zur Extraktion wirklich relevanter Sätze beigetragen haben, denn bei Vernachlässigung dieser Komponenten sind sowohl Recall als auch Precision am kleinsten. Auch bei der bloßen Beachtung der Suchworte (siehe Zusammenfassung 11) werden gute Ergebnisse erzielt. Allerdings sind die Recall- und Precisionwerte im Allgemeinen geringer als beim vorherigen Text. Meist sind bei längeren Texten bessere Werte zu erwarten.

Zusammenfassung	Ausgewählte Sätze	Recall	Precision
1	1, 4, 10, 22	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
2	1, 4, 10, 22	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
3	4, 6, 17, 22	0	0
4	1, 4, 7, 22	0	0
5	1, 4, 10, 22	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
6	10, 22, 23, 24	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
7	1, 4, 10, 22	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
8	1, 4, 6, 8	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
9	1, 4, 7, 10	$\frac{2}{6} = 0{,}333$	$\frac{2}{4} = 0{,}5$
10	6, 10, 17, 21	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
11	1, 4, 17, 18	0	0

Tabelle 15: Zusammenfassungen des dritten Textes mit dem Satzgewicht-Algorithmus und dazugehörige Recall- und Precisionwerte

Die Recall- und Precisionwerte, die bei der Zusammenfassung des letzten Textes erzielt wurden, sind sehr gering und sogar mehrfach gleich Null. Dabei konnten die besten Werte bei der bloßen Beachtung der Satzposition erreicht werden. Anders als in den beiden vorangegangenen Texten konnten bei der bloßen Berücksichtigung der Suchworte nur sehr geringe Recall- und Precisionwerte berechnet werden.

4.3.5.2.4. Zusammenfassungen mit dem Genetischen Algorithmus

Mit dem Genetischen Algorithmus ließen wir die drei Texte jeweils zehnmal mit unterschiedlichen Gewichten zusammenfassen.

Zusammenfassung	1	2	3	4	5	6	7	8	9	10
$Gewicht_{Satzposition}$	1	0	1	1	1	1	0	1	0	0
$Gewicht_{Satzlänge}$	1	1	0	1	1	1	0	0	1	0
$Gewicht_{Abschnitt}$	1	1	1	0	1	1	0	0	0	0
$Gewicht_{enth.Suchworte}$	1	1	1	1	0	1	1	0	0	1
$Gewicht_{enth.SuchworteInÜberschr}$	1	1	1	1	0	1	1	0	0	0
$Gewicht_{wicht.Phrasen}$	1	1	1	1	1	0	0	0	0	0
$Gewicht_{unwicht.Phrasen}$	-1	-1	-1	-1	-1	0	0	0	0	0
$Gewicht_{Fragen}$	-1	-1	-1	-1	-1	-1	0	0	0	0

Tabelle 16: Komponentengewichte der einzelnen Zusammenfassungen

Die Suchworte für die verschiedenen Texte sind dieselben wie bei den vorherigen Algorithmen:

- Text1: Ayla Urteil
- Text2: Vogelgrippe Türkei tot
- Text3: SPD Klausur Ziel

Zusammenfassung	Ausgewählte Sätze	Recall	Precision
1	3, 5, 12, 13, 15	$\frac{2}{7} = 0{,}286$	$\frac{2}{5} = 0{,}4$
2	2, 10, 18, 21, 23	0	0
3	1, 8, 12, 16, 18	$\frac{2}{7} = 0{,}286$	$\frac{2}{5} = 0{,}40$
4	1, 5, 16, 17, 22	$\frac{3}{7} = 0{,}429$	$\frac{3}{5} = 0{,}6$
5	5, 16, 19, 21, 22	$\frac{3}{7} = 0{,}429$	$\frac{3}{5} = 0{,}6$
6	2, 12, 14, 16, 17	$\frac{1}{7} = 0{,}143$	$\frac{1}{5} = 0{,}2$
7	1, 6, 11, 15a, 17	$\frac{1}{7} = 0{,}143$	$\frac{1}{5} = 0{,}2$
8	3, 7, 12, 16, 19	$\frac{4}{7} = 0{,}571$	$\frac{4}{5} = 0{,}8$
9	3, 5, 13, 16, 20	$\frac{3}{7} = 0{,}429$	$\frac{3}{5} = 0{,}6$

10	1, 11, 13, 15, 15a	$\frac{1}{7} = 0{,}143$	$\frac{1}{5} = 0{,}2$

Tabelle 17: Zusammenfassungen des ersten Textes mit dem Genetischen Algorithmus und dazugehörige Recall- und Precisionwerte

An den Precision- und Recallwerten der Zusammenfassungen des ersten Textes kann man erkennen, dass diese Werte sehr stark schwanken (von 0 bis 0,8 bzw. von 0 bis 0,571). Auch die ausgewählten Sätze der einzelnen Summaries unterscheiden sich sehr stark von einander.

Zusammenfassung	Ausgewählte Sätze	Recall	Precision
1	9, 10, 15, 17, 18, 30, 33	$\frac{2}{10} = 0{,}2$	$\frac{2}{7} = 0{,}286$
2	6, 8, 14, 22, 28, 30, 36	$\frac{2}{10} = 0{,}2$	$\frac{2}{7} = 0{,}286$
3	1, 15, 16, 18, 19, 25, 29	$\frac{2}{10} = 0{,}2$	$\frac{2}{7} = 0{,}286$
4	10, 12, 15, 23, 26, 33, 35	$\frac{3}{10} = 0{,}3$	$\frac{3}{7} = 0{,}429$
5	4, 7, 23, 30, 31, 33, 34	$\frac{1}{10} = 0{,}1$	$\frac{1}{7} = 0{,}143$
6	10, 12, 16, 17, 23, 25, 27	$\frac{2}{10} = 0{,}2$	$\frac{2}{7} = 0{,}286$
7	2, 11, 20, 21, 27, 28, 29	$\frac{2}{10} = 0{,}2$	$\frac{2}{7} = 0{,}286$
8	1, 9, 17, 18, 21, 23, 29	$\frac{4}{10} = 0{,}4$	$\frac{4}{7} = 0{,}571$
9	6, 15, 18, 23, 27, 30, 36	$\frac{2}{10} = 0{,}2$	$\frac{2}{7} = 0{,}286$
10	1, 6, 11, 20, 22, 25, 27	$\frac{2}{10} = 0{,}2$	$\frac{2}{7} = 0{,}286$

Tabelle 18: Zusammenfassungen des zweiten Textes mit dem Genetischen Algorithmus und dazugehörige Recall- und Precisionwerte

Die Zusammenfassungen des zweiten Textes, die mit dem Genetischen Algorithmus erstellt wurden, weisen durchweg geringe Recall- und Precisionwerte auf (mit Ausnahme der 8. Zusammenfassung). Allerdings gibt es nie Recall- oder Precisionwerte, die gleich Null sind, das heißt, dass stets mindestens ein relevanter Satz Teil der Zusammenfassungen ist.

Zusammenfassung	Ausgewählte Sätze	Recall	Precision
1	6, 17, 19, 22	0	0
2	8, 10, 17, 24	$\frac{2}{6} = 0{,}333$	$\frac{2}{4} = 0{,}5$
3	1, 2, 5, 23	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
4	17, 19, 21, 23	0	0
5	2, 3, 17, 23	0	0
6	9, 19, 21, 22	0	0
7	4, 15, 16, 17	0	0
8	1, 2, 5, 23	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
9	1, 2, 11, 23	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
10	1, 2, 15, 23	0	0

Tabelle 19: Zusammenfassungen des dritten Textes mit dem Genetischen Algorithmus und dazugehörige Recall- und Precisionwerte

Beim dritten Text sind die Ergebnisse noch schlechter. Nur in vier der zehn Zusammenfassungen waren überhaupt relevante Sätze enthalten. Des Weiteren kann man auch in der Auswahl der Sätze kein System und keinen Bezug zu den unterschiedlichen Gewichten erkennen.

4.3.5.2.5. Zusammenfassungen mit unserem Algorithmus

Für das Erstellen von Zusammenfassungen mit unserem eigenen Algorithmus gibt es sechs Parameter, mit denen dieser Prozess beeinflusst werden kann. Die zwei Parameter „mit Bildern“ und „Berücksichtigung von Links“ werden allerdings nur bei der Zusammenfassung von HTML-Seiten wirksam. Daher entschieden wir uns pro Text neun Summaries generieren zu lassen mit folgenden Parametern:

Zusammenfassung	PositionsWert	ÜberschriftsinhaltWert	SuchwortWert	EigennamenWert
1	1	1	1	1
2	0	1	1	1
3	1	0	1	1
4	1	1	0	1
5	1	1	1	0
6	1	0	0	0
7	0	1	0	0
8	0	0	1	0
9	0	0	0	1

Tabelle 20: Parametrisierung der neun Zusammenfassungen

Die Suchworte waren dieselben wie bei den vorherigen Algorithmen.

Zusammenfassung	Ausgewählte Sätze	Recall	Precision
1	1, 3, 5, 6, 21	$\frac{3}{7} = 0{,}429$	$\frac{3}{5} = 0{,}6$
2	1, 3, 5, 6, 21	$\frac{3}{7} = 0{,}429$	$\frac{3}{5} = 0{,}6$
3	1, 3, 4, 6, 21	$\frac{3}{7} = 0{,}429$	$\frac{3}{5} = 0{,}6$
4	1, 3, 5, 6, 21	$\frac{3}{7} = 0{,}429$	$\frac{3}{5} = 0{,}6$
5	1, 3, 5, 6, 16	$\frac{4}{7} = 0{,}571$	$\frac{4}{5} = 0{,}8$
6	1, 2, 3, 6, 11	$\frac{2}{7} = 0{,}289$	$\frac{2}{5} = 0{,}4$
7	1, 3, 5, 6, 22	$\frac{3}{7} = 0{,}429$	$\frac{3}{5} = 0{,}6$
8	1, 4, 5, 6, 11	$\frac{3}{7} = 0{,}429$	$\frac{3}{5} = 0{,}6$
9	3, 6, 11, 18, 21	$\frac{1}{7} = 0{,}143$	$\frac{1}{5} = 0{,}2$

Tabelle 21: Zusammenfassungen des ersten Textes mit unserem Algorithmus und dazugehörige Recall- und Precisionwerte

Die Zusammenfassungen, welche unser Algorithmus aus dem ersten Text erstellt hat, haben im Vergleich mit den vorangegangenen Versuchen relativ gute Recall- und Precisionwerte. So sind in der fünften Zusammenfassung immerhin 80% aller Sätze relevante Sätze. Die schlechtesten Werte wurden bei der bloßen Berücksichtigung von Eigennamen erzielt.

Zusammenfassung	Ausgewählte Sätze	Recall	Precision
1	1, 2, 6, 8, 11, 26, 32	$\frac{5}{10} = 0{,}5$	$\frac{5}{7} = 0{,}714$
2	1, 2, 4, 6, 8, 11, 26	$\frac{5}{10} = 0{,}5$	$\frac{5}{7} = 0{,}714$
3	1, 2, 4, 6, 8, 26, 32	$\frac{5}{10} = 0{,}5$	$\frac{5}{7} = 0{,}714$
4	8, 11, 16, 23, 24, 26, 32	$\frac{3}{10} = 0{,}3$	$\frac{3}{7} = 0{,}429$
5	1, 2, 6, 8, 11, 23, 26	$\frac{6}{10} = 0{,}6$	$\frac{6}{7} = 0{,}857$
6	8, 22, 23, 26, 28, 29, 32	$\frac{4}{10} = 0{,}4$	$\frac{4}{7} = 0{,}571$
7	1, 6, 8, 11, 14, 23, 26	$\frac{5}{10} = 0{,}5$	$\frac{5}{7} = 0{,}714$
8	1, 2, 6, 8, 11, 23, 26	$\frac{6}{10} = 0{,}6$	$\frac{6}{7} = 0{,}857$
9	4, 6, 8, 16, 23, 26,32	$\frac{4}{10} = 0{,}4$	$\frac{4}{7} = 0{,}571$

Tabelle 22: Zusammenfassungen des zweiten Textes mit unserem Algorithmus und dazugehörige Recall- und Precisionwerte

Auch die Zusammenfassungen des zweiten Textes erzielen fast alle gute bis sehr gute Werte für Recall und Precision. Nur das Summary, bei dem die eingegebenen Suchworte nicht berücksichtigt werden, hat Recall- und Precisionwerte im unteren Bereich.

Zusammenfassung	Ausgewählte Sätze	Recall	Precision
1	1, 4, 17, 22	0	0
2	1, 4, 17, 18	0	0
3	4, 8, 17, 22	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
4	4, 19, 21, 22	0	0
5	1, 4, 17, 18	0	0
6	10, 17, 20, 22	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
7	1, 4, 11, 17	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
8	1, 4, 8, 17	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$
9	4, 5, 15, 21	$\frac{1}{6} = 0{,}167$	$\frac{1}{4} = 0{,}25$

Tabelle 23: Zusammenfassungen des dritten Textes mit unserem Algorithmus und dazugehörige Recall- und Precisionwerte

Entgegen dem Trend, der sich bei den beiden vorangegangenen Texten abgezeichnet hat, sind die Zusammenfassungen des dritten Textes zum großen Teil unbrauchbar, da bei vier Summaries nicht ein relevanter Satz enthalten ist. Die restlichen Zusammenfassungen enthalten alle nur einen relevanten Satz.

4.3.5.2.6. Zusammenfassungen mit Word

Wir ließen mit der AutoZusammenfassen-Funktion von Microsoft® Office Word 2003 die drei Texte so zusammenfassen, dass einmal 20%ige Zusammenfassungen generiert wurden und dass außerdem Zusammenfassungen mit der gleichen Anzahl an Sätzen wie bei den vorherigen Algorithmen erstellt werden. Das die 20%igen Summaries nicht automatisch so groß sind, wie die 20%igen Zusammenfassungen, die unser Zusammenfassungsprogramm erzeugt hat, liegt daran, dass bei Word die Zusammenfassung nicht 20% der Gesamtsatzanzahl als Länge hat, sondern dass 20% der Gesamtwortanzahl als Größe genommen wird.

Zusammenfassung	Ausgewählte Sätze	Recall	Precision
20%	1, 12, 13, 19	$\frac{2}{7} = 0{,}289$	$\frac{2}{4} = 0{,}5$
5 Sätze	1, 8, 12, 13, 19	$\frac{2}{7} = 0{,}289$	$\frac{2}{5} = 0{,}4$

Tabelle 24: Zusammenfassungen des ersten Textes mit dem Word-Zusammenfasser und dazugehörige Recall- und Precisionwerte

Die Recallwerte für die Zusammenfassungen des ersten Textes liegen im unteren Bereich, während die Precisionwerte im mittleren Bereich liegen.

Zusammenfassung	Ausgewählte Sätze	Recall	Precision
20%	1, 3, 4, 8, 10, 19, 24, 32	$\frac{3}{10} = 0{,}3$	$\frac{3}{8} = 0{,}375$
5 Sätze	1, 3, 4, 8, 10, 19, 24	$\frac{3}{10} = 0{,}3$	$\frac{3}{7} = 0{,}429$

Tabelle 25: Zusammenfassungen des zweiten Textes mit dem Word-Zusammenfasser und dazugehörige Recall- und Precisionwerte

Die erzielten Recallwerte sind zusammen mit denen des Luhn-Algorithmus die zweitschlechtesten. Auch die Präzision ist schlechter als die der meisten anderen Algorithmen mit Ausnahme des Genetischen Algorithmus.

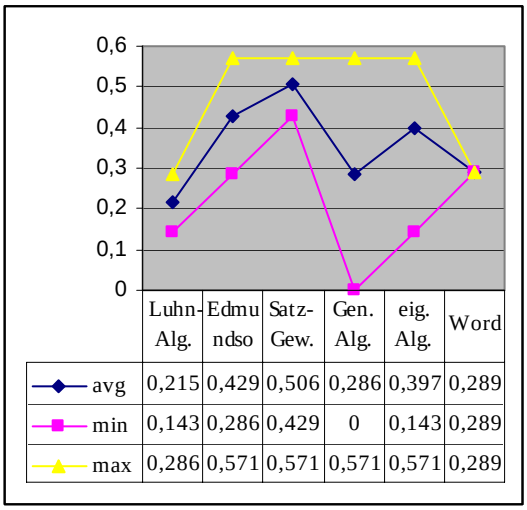
Zusammenfassung	Ausgewählte Sätze	Recall	Precision
20%	1, 3, 4, 14, 22	0	0
5 Sätze	1, 3, 4, 22	0	0

Tabelle 26: Zusammenfassungen des dritten Textes mit dem Word-Zusammenfasser und dazugehörige Recall- und Precisionwerte

Bei der Zusammenfassung des letzten Textes wählte der Word-Zusammenfasser nicht einen einzigen relevanten Satz aus und erzielte damit das schlechteste Resultat.

4.3.5.3. Auswertung der Ergebnisse

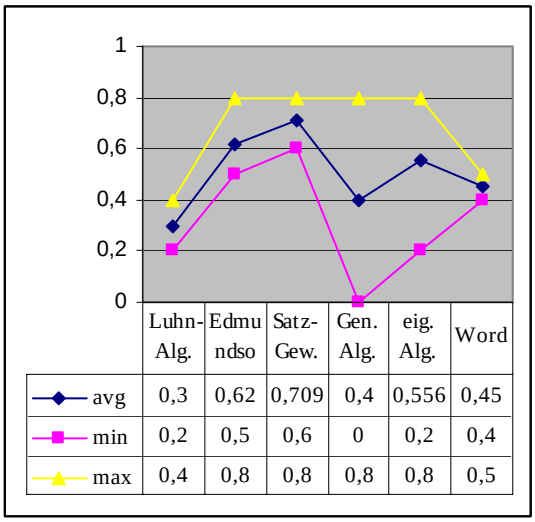
Recall – Werte



	Luhn-Alg.	Edmundso	Satz-Gew.	Gen. Alg.	eig. Alg.	Word
avg	0,215	0,429	0,506	0,286	0,397	0,289
min	0,143	0,286	0,429	0	0,143	0,289
max	0,286	0,571	0,571	0,571	0,571	0,289

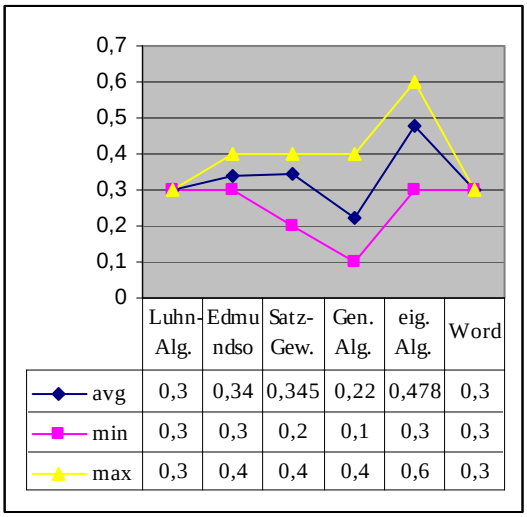
Abbildung 50: durchschn., max. und min. Recall-Werte für die unterschiedlichen Algorithmen bei der Zusammenfassung des ersten Textes

Precision – Werte



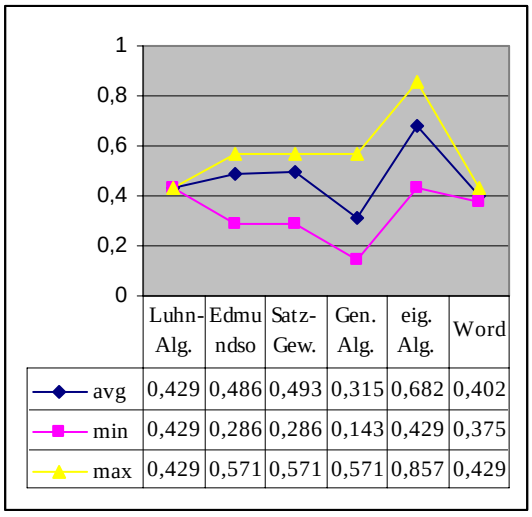
	Luhn-Alg.	Edmundso	Satz-Gew.	Gen. Alg.	eig. Alg.	Word
avg	0,3	0,62	0,709	0,4	0,556	0,45
min	0,2	0,5	0,6	0	0,2	0,4
max	0,4	0,8	0,8	0,8	0,8	0,5

Abbildung 51: durchschn., max. und min. Precision-Werte für die unterschiedlichen Algorithmen bei der Zusammenfassung des ersten Textes



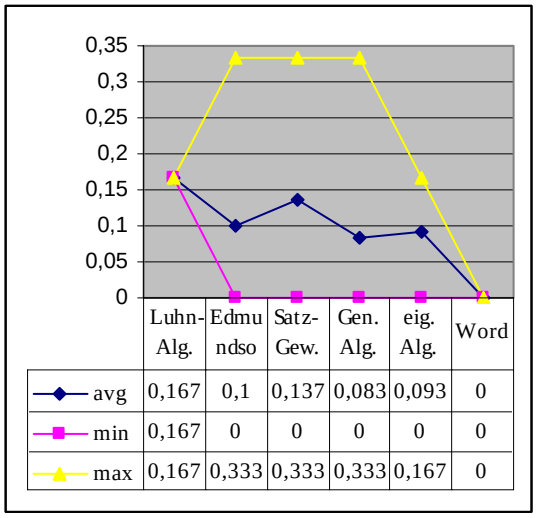
	Luhn-Alg.	Edmundso	Satz-Gew.	Gen. Alg.	eig. Alg.	Word
avg	0,3	0,34	0,345	0,22	0,478	0,3
min	0,3	0,3	0,2	0,1	0,3	0,3
max	0,3	0,4	0,4	0,4	0,6	0,3

Abbildung 52: durchschn., max. und min. Recall-Werte für die unterschiedlichen Algorithmen bei der Zusammenfassung des zweiten Textes



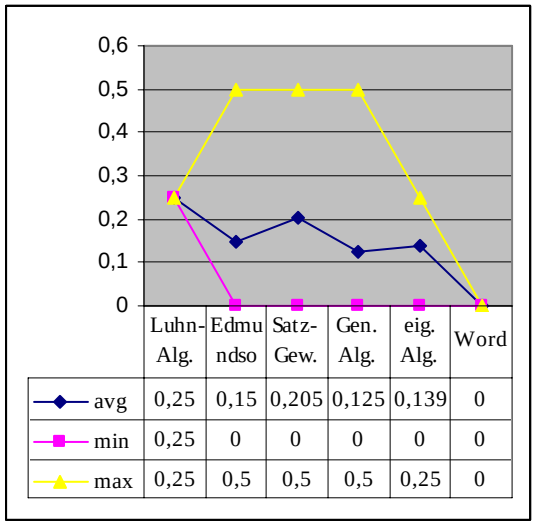
	Luhn-Alg.	Edmundso	Satz-Gew.	Gen. Alg.	eig. Alg.	Word
avg	0,429	0,486	0,493	0,315	0,682	0,402
min	0,429	0,286	0,286	0,143	0,429	0,375
max	0,429	0,571	0,571	0,571	0,857	0,429

Abbildung 53: durchschn., max. und min. Precision-Werte für die unterschiedlichen Algorithmen bei der Zusammenfassung des zweiten Textes



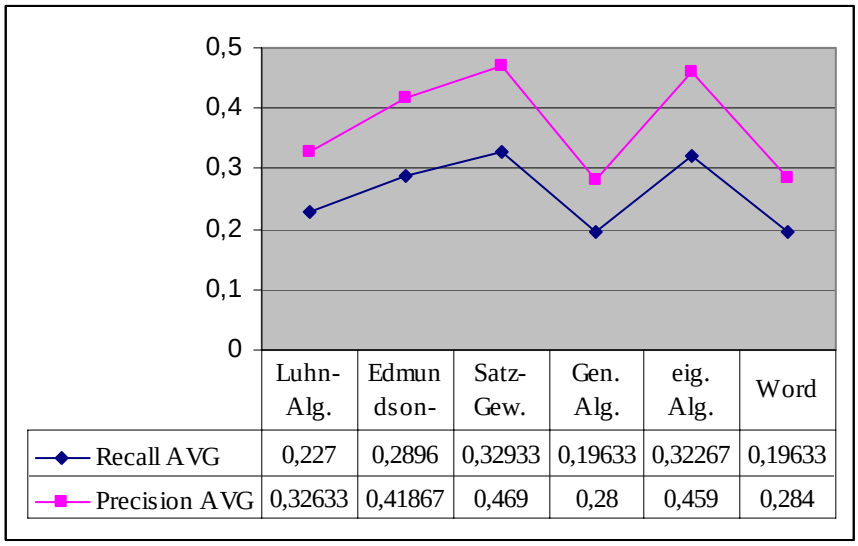
	Luhn-Alg.	Edmundso	Satz-Gew.	Gen. Alg.	eig. Alg.	Word
avg	0,167	0,1	0,137	0,083	0,093	0
min	0,167	0	0	0	0	0
max	0,167	0,333	0,333	0,333	0,167	0

Abbildung 54: durchschn., max. und min. Recall-Werte für die unterschiedlichen Algorithmen bei der Zusammenfassung des dritten Textes



	Luhn-Alg.	Edmundso	Satz-Gew.	Gen. Alg.	eig. Alg.	Word
avg	0,25	0,15	0,205	0,125	0,139	0
min	0,25	0	0	0	0	0
max	0,25	0,5	0,5	0,5	0,25	0

Abbildung 55: durchschn., max. und min. Precision-Werte für die unterschiedlichen Algorithmen bei der Zusammenfassung des dritten Textes



	Luhn-Alg.	Edmundson-Alg.	Satz-Gew.	Gen. Alg.	eig. Alg.	Word
Recall AVG	0,227	0,2896	0,32933	0,19633	0,32267	0,19633
Precision AVG	0,32633	0,41867	0,469	0,28	0,459	0,284

Abbildung 56: Durchschnittlich Recall- u. Precisionwerte aller drei Texte für alle Algorithmen

Die interessanteste Linie in den Abbildungen 50 bis 55 ist die Avg-Linie. Sie zeigt die durchschnittlichen Recall- und Precisionwerte der verschiedenen Algorithmen mit unterschiedlichen Parametern in den drei Texten. Wie man an dieser Linie erkennen kann, waren die Zusammenfassungen des ersten Textes, die mit dem Satzgewicht-Algorithmus erstellt wurden, die besten. Bei dem zweiten Text schnitten die Zusammenfassungen am besten ab, welche unser Algorithmus generiert hat. Bei der Zusammenfassung des dritten Textes, konnten die Summaries des Luhn-Algorithmus die höchsten durchschnittlichen Precision- und Recallwerte erreichen.

Der Luhn-Algorithmus, welcher nur auf der Analyse der Worthäufigkeiten basiert, lieferte beim ersten Text die schlechtesten Zusammenfassungen. Im zweiten Text waren die mit ihm erzeugten Zusammenfassungen zusammen mit denen des Word-Zusammenfassers die zweitschlechtesten und im dritten Text waren seine Zusammenfassungen mit geringeren Precision- und Recallwerten als bei den vorangegangenen Texten die besten. Diese Anomalie liegt wahrscheinlich an der Referenzzusammenfassung des dritten Textes, denn bei diesem Text haben alle Algorithmen viel kleinere Recall- und Precisionwerte erreicht als zuvor. So wurden von den Teilnehmern der Evaluierung zum Beispiel die Sätze 1 – 3, welche eine kurze Zusammenfassung des Themas am Anfang des Textes bieten, nur sehr selten markiert und daher nicht in die Referenzzusammenfassung übernommen. Von den meisten Algorithmen wurden diese Sätze aber als relevant markiert. Die Teilnehmer der Evaluation legten ihr Hauptaugenmerk in diesem Text auf die Themen „Kinderbetreuung“ und „Kindergarten“. Hätte man vielleicht diese Worte den Algorithmen als Suchworte zur

besseren Berücksichtigung von Nutzerinteressen geboten, würden die Zusammenfassungen mit großer Wahrscheinlichkeit besser werden.

Im Großen und Ganzen würde ich daher den Luhn-Algorithmus trotz des guten Abschneidens beim dritten Text als den zweitschlechtesten Algorithmus bezeichnen. Auch die unterschiedlichen Methoden zur Bestimmung der relevanten Worte hatten keine Auswirkungen auf die Qualität der Zusammenfassungen. Des Weiteren bietet er keine Möglichkeiten auf Nutzerinteressen oder Texteigenschaften einzugehen.

Der Zusammenfasser des kommerziellen Textverarbeitungsprogramms Word und der Genetische Algorithmus, welcher bei den Zusammenfassungen des zweiten Textes die schlechtesten und beim dritten Text die zweitschlechtesten Werte erzielte, sind meiner Meinung nach die schlechtesten Algorithmen dieser Testreihe. Bei den Summaries des dritten Textes erstellte der Word-Zusammenfasser mit Abstand die schlechtesten Zusammenfassungen, in denen nicht ein relevanter Satz enthalten war. Auch bei den Zusammenfassungen der anderen beiden Texte konnte er nur die zweitschlechtesten bzw. drittschlechtesten Ergebnisse erzielen. Allerdings muss erwähnt werden, dass durch das Nicht-Vorhandensein einer Möglichkeit zur Parametrisierung mit dem Word-Zusammenfasser auch die geringste Anzahl an Zusammenfassungen pro Text erstellt wurden, so dass die Recall- und Precisionwerte nicht ganz so aussagekräftig sind wie bei den anderen Algorithmen, bei denen mindestens vier Zusammenfassungen pro Text erstellt wurden.

Der Ansatz des Genetischen Algorithmus, welcher aus der Evolution der Natur abgeschaut ist, ist eigentlich interessant. Allerdings legt sich der Algorithmus sehr schnell auf ein bestimmtes Zusammenfassungsindividuum (d.h. eine bestimmte Zusammenfassung) fest, welche sich dann über die restlichen Evolutionsschritte nicht mehr ändert. Ein weiteres Problem stellen die zufällig generierten Anfangszusammenfassungen dar. Dabei werden Anfangsindividuen aus zufällig ausgewählten Sätzen generiert. Diese Anfangspopulationen als Ausgangspunkt der Evolution verändern sich aber durch Mutation, Selektion und Vererbung nicht so wie erwartet. Das heißt, es bleiben am Ende zwar die besten Zusammenfassungen der Population übrig, dies sind aber nicht die bestmöglichen. Vielleicht würde eine andere Initialisierung der Population, welche eventuell nicht ganz zufällig ist, eine Verbesserung bringen. So könnten zum Beispiel Sätze an bestimmten Positionen bevorzugt in diese Anfangsindividuen aufgenommen werden. Eine andere Möglichkeit wäre zum Beispiel noch mehr Startzusammenfassungen am Anfang zu generieren, da so die Wahrscheinlichkeit größer ist, dass unter den vielen möglichen Kombinationen von Sätzen auch die beste dabei ist. Das würde aber den Aufwand und die Zeit zur Erstellung einer Zusammenfassung gerade bei

längeren Texten stark erhöhen und den eigentlichen Vorteil dieses Algorithmus, dass nicht für jeden Satz das Gewicht berechnet werden muss, fast außer Kraft setzen, so dass man sich dann fragen muss, warum man nicht doch für alle Sätze erst das Gewicht berechnet und sich dann daraus die besten Sätze auswählt.
Meiner Meinung nach geht aus dieser Evaluierung der Satzgewicht-Algorithmus als Sieger hervor, was man auch in Abbildung 56 gut erkennen kann. Er erstellte für den ersten Text die besten und für den zweiten Text die zweitbesten Zusammenfassungen. Sehr gut erkennbar ist auch, dass die Berücksichtigung der Suchworte die Ergebnisse dieses Algorithmus entscheidend verbessert hat. Während bei der Gleichgewichtung aller Komponenten Recall- und Precisionwerte von 0.571 und 0.8 beim ersten Text erzielt werden, sinken Recall und Precision auf 0.429 und 0.6 ab, als das Gewicht für die Suchworte gleich Null ist. Die erzielten Ergebnisse können meiner Meinung nach noch verbessert werden, wenn man für die Gewichtung der einzelnen Komponenten mit Hilfe vieler Tests die richtigen Werte ermittelt. Wir haben die Werte recht willkürlich gewählt und nur die Werte 0 und 1 ausgewählt, um einzelne Komponenten zu berücksichtigen bzw. ganz zu vernachlässigen. Auffällig ist auch, dass die unterschiedlichen Gewichte sich gerade im ersten Text kaum auf die Auswahl der Sätze auswirken. Allerdings waren die Beispieltexte alle recht kurz, so dass ein Test mit längeren Texten sicher eine bessere Differenzierung zwischen den unterschiedlichen Gewichtungen bringen dürfte. Diese Tendenz ist bei den Zusammenfassungen des zweiten Textes, welcher etwas länger war, schon erkennbar.

Den Edmundson-Algorithmus würde ich als drittbesten Algorithmus dieser Evaluierung bezeichnen. Unser Algorithmus hat bei der Zusammenfassung des zweiten Textes entscheidend besser abgeschnitten als der Edmundson-Algorithmus. Zwar ist dieser Algorithmus bei den anderen beiden Texten etwas besser als unser eigener, aber wenn man das Diagramm mit den Gesamtdurchschnitten von Recall und Precision betrachtet (siehe Abbildung 56), dann ist ersichtlich, dass unser Algorithmus bessere Werte erzielen konnte.
Bei den Ergebnissen des Edmundson-Algorithmus ist erkennbar, dass hauptsächlich die Key- und die Title-Methode für die guten Recall- und Precisionwerte verantwortlich sind. Beim ersten und beim zweiten Text haben nämlich die bloße Berücksichtung der Keymethode bzw. der Titlemethode die gleichen Recall- und Precisionwerte wie die Zusammenfassung, die aller vier Methoden gleich berücksichtigt. Ein Grund für die schlechten Ergebnisse bei der bloßen Berücksichtigung der Locationmethode könnten die Überschriften im Headingdictionary sein. Diese Überschriften findet man eher in wissenschaftlichen Texten. Da es sich bei unseren

Texten um Zeitungsartikel handelte, ist zu erwarten, dass Überschriften wie „Einleitung“, „Einführung“ oder „Ergebnis“ nicht vorkommen und daher diese Methode die Sätze nicht entsprechend ihrer Wichtigkeit differenzieren kann. Eine ähnliche Begründung lässt sich auch für die schlechteren Recall- und Precisionwerte der Cuemethode finden. Ebenso wie die Überschriften im Headingdictionary so sind auch die Bonus- und Stigmaworte des Cuedictionary eher auf wissenschaftliche Texte ausgerichtet. Worte wie „signifikant“, „abschließend“ oder „unwichtig“ sind in Zeitungsartikeln selten zu finden und würden dann wahrscheinlich auch nicht unbedingt auf relevante bzw. irrelevante Sätze hinweisen. Durch das Nicht-Vorhandensein dieser Cueworte konnte die Cuemethode wahrscheinlich keine Satzgewichte ungleich Null für alle Sätze ermitteln, so dass alle Sätze das gleiche Gewicht hatten und dadurch einfach die ersten n Sätze für die Zusammenfassungen ausgewählt wurden (vgl. Tabelle 9 und 10). Dass gerade diese beiden Methoden schlechte Ergebnisse erzielen war vorhersehbar, da hier die linguistische Quelle des Hinweises auf einen relevanten Satz die allgemeinen Eigenschaften des Textkorpus bzw. der Textart sind (vgl. Abbildung 3).

Dieser Algorithmus kann allerdings nicht mit Hilfe von Suchworten an die Interessen des Nutzers angepasst werden.

Die Ergebnisse unseres Algorithmus bei der Zusammenfassung des zweiten Textes sind sehr gut. Durchschnittlich waren 65% aller ausgewählten Sätze auch relevant und immerhin 48% der relevanten Sätze wurden extrahiert. Im besten Fall waren sogar 85% der Sätze des Summary relevant und 60% der relevanten Sätze wurden ausgewählt. Dabei muss erwähnt werden, dass maximal 70% der relevanten Sätze überhaupt für eine Zusammenfassung ausgewählt werden konnten, da 10 Sätze als relevant deklariert worden waren und bei einer 20%-Zusammenfassung nur 7 Sätze ausgewählt werden durften. Auch zeigt sich gerade im ersten Text sehr deutlich, dass die Berücksichtigung von Eigennamen nicht den gewünschten Effekt brachte. Zum einen brachte die Parametrisierung, bei der der Eigennamenwert gleich Null ist, die beste Zusammenfassung hervor und zum anderen war die Zusammenfassung, die die Sätze nur nach den Eigennamen gewichtet hatte, die schlechteste. Ein Grund dafür könnte das Thema des Textes sein, da hier im Interesse nicht namentlich erwähnte Personen standen sondern das verhängte Urteil und die Straftat. Im zweiten Text ist dieses Phänomen nicht mehr beobachtbar, da hier in den relevanten Sätzen die Namen von Personen erwähnt wurden. Ein weiteres Problem könnte aber auch sein, dass das Programm nicht nur wirkliche Eigennamen als solche erkennt. So werden Wortgruppen wie „Schwester Hülya“, „Familie Kocyigit“, „Großstadt Van“ oder „Skorpion-Stichen“ als Eigennamen bewertet. Auch ist es nicht möglich Personen, die nur mit dem Vornamen erwähnt werden, als Eigennamen zu

erkennen. Gute Ergebnisse wurden auch erzielt, als die Sätze nur nach dem Vorkommen von Suchworten bewertet wurden. Aber auch wenn solche Vorgaben wie Suchworte nicht vorhanden sind, konnten allein mit der Beachtung des Inhalts der Überschriften recht gute Zusammenfassungen generiert werden. Wichtig zu erwähnen ist außerdem, dass man auch mit diesem Algorithmus eventuell noch bessere Ergebnisse erzielen könnte, wenn man die vier Parameter anders gewichtet. Allerdings wären dafür viele Tests mit vielen Texten und unterschiedlichster Parametrisierung von Nöten, um die beste Kombination zu ermitteln.

1	Satzgewicht-Algorithmus
2	Eigener Algorithmus
3	Edmundson-Algorithmus
4	Luhn-Algorithmus
5	Word-Zusammenfasser und Genetischer Algorithmus

Abbildung 57: erzielte Platzierung nach der Evaluation

5. Zusammenfassung

Das Ziel dieser Arbeit war es, Satzextraktionsalgorithmen zu analysieren, einige davon zu implementieren und die Ergebnisse dieser Zusammenfassungsalgorithmen zu vergleichen. Wir analysierten dabei Algorithmen aus den Anfängen der Forschung, wie die Algorithmen von Luhn und Edmundson, aber auch Algorithmen aus den 1990er Jahren wurden von uns bezüglich ihrer Satzauswahlmethoden untersucht. Des Weiteren befassten wir uns ebenfalls mit aktuellen Forschungsarbeiten auf dem Gebiet der Automatischen Textzusammenfassung, wie beispielsweise mit der Zusammenfassung durch Satzreduktion oder der Satzauswahl durch Stichwortlisten. Aus diesen analysierten Algorithmen wählten wir dann fünf Algorithmen, welche Zusammenfassungen durch gezielte Satzextraktionen erstellen und welche implementiert und getestet werden konnten, ohne dass spezielle Datenbanken nötig waren, ohne dass riesige Testkorpora benötigt wurden und ohne dass die Extraktionsregeln in langwierigen Testläufen erst erlernt werden mussten. Der Algorithmus von Euler wurde trotz des eigentlich für die Erstellung der Stichwortlisten benötigten Textkorpus gewählt, da dieser viel versprechend klang und von einem Studenten entwickelt worden war. Zu Testzwecken hatten wir dafür von Hand mit Hilfe einiger weniger Texte für eine bestimmte Domäne eine Stichwortliste erstellt. Allerdings konnte dieser Algorithmus in der Evaluierung nicht berücksichtigt werden, da die erstellte Stichwortliste nicht zu einem der drei Themen der Versuchstexte passte und die Generierung von solchen Listen für alle drei Texte zu aufwendig gewesen wäre.

Des Weiteren implementierten wir einen eigenen Algorithmus, der sich teilweise an dem Satzgewichtalgorithmus orientierte, aber weitere Kriterien bei der Satzauswahl berücksichtigte.

Als im Evaluierungsteil dieser Arbeit die fünf verschiedenen Algorithmen an drei unterschiedlichen Texten getestet werden sollten, entschieden wir uns für eine intrinsische Evaluierung, welche die erstellten Zusammenfassungen mit einer Referenzzusammenfassung verglich. Diese Referenzzusammenfassungen wurden aus den Ergebnissen einer Studentenbefragung erstellt, bei der die Teilnehmer in den drei Testtexten wichtige Sätze markieren sollten. Als Resultat der Evaluation ging der Satzgewicht-Algorithmus aus diesem Vergleich als Sieger hervor. Unser Extraktionsalgorithmus belegte mit nur geringfügig schlechteren Werten den zweiten Platz. Der kommerzielle Zusammenfasser von Word lieferte zusammen mit dem Genetischen Algorithmus die schlechtesten Zusammenfassungen.

Es zeigte sich als Gesamtresultat, dass man mit der bloßen Auswertung von Eigenschaften der Sätze oder der Worte schon recht gute Aussagen über die Wichtigkeit eines Satzes machen kann und so gezielt die relevanten Sätze für eine Zusammenfassung auswählen kann. Allerdings gab es bei der Implementierung auch einige Probleme.
Unser Algorithmus, welcher schon recht gute Ergebnisse erzielt hat, könnte durch verschiedene Verbesserungsvorschläge oder Erweiterungen noch optimiert werden.

5.1. Probleme bei der Implementierung

Bei der Implementierung des Zusammenfassungsprogramms gab es einige verschiedene Probleme.
Ein größeres Problem war das Einlesen der Texte in den unterschiedlichen Formaten. Es stellte sich zum Beispiel als schwieriger als erwartet heraus, die Überschriften zu erkennen. Während die ganze Problematik bei HTML-Dokumenten durch die h-Tags sehr einfach war, war es bei RTF-Dokumenten sehr viel schwerer. Obwohl wir durch unsere zwei Regeln, alle Abschnitte ohne Punkt und alle Abschnitte, die mit einer Nummerierung beginnen, sind Überschriften, schon sehr gut Überschriften herausfilterten, wurden auch Beispiele, die aus kurzen Wortgruppen bestanden, oder Bildunterschriften oft fälschlich als Überschrift deklariert. Hier ein Beispiel:

> *„Der Grund hierfür ist einfach der, daß im Morsealphabet einige Kodes auch den Anfang anderer Kodes bilden.*
>
> ```
>  E        .
>  I        ..
>  N        _.
>  B        _...
>
> NEIN    _. . .. _.
> BEN      _... . _.
> ```
>
> ***Auszug aus dem Morsealphabet, Beispiele NEIN, BEN***
>
> *Das Morsealphabet läßt sich als Binärbaum beschreiben, in dem (fast) jedem Knoten außer der Wurzel ein Buchstabe zugeordnet wird, jedem Linksabstieg ein kurzes Signal, jedem Rechtsabstieg ein langes*.“ [Katz 2002]

Da in der Bildunterschrift kein Satzpunkt zu finden ist, nimmt das Programm an, dass es sich um eine Überschrift handelt. Denkbar wäre eventuell eine genaue Analyse der RTF-Tags, allerdings muss dafür gewährleistet sein, dass Überschriften in RTF-Dokumenten auch als solche deklariert werden.

Ein weiteres Problem war die Erkennung von Sätzen. In RTF-Dokumenten wurde ein Abschnitt an möglichen Satzendzeichen („. “, „! “, „? “) in einzelne Sätze zerlegt. Da allerdings ein Punkt nicht unbedingt ein Satzende markiert, überprüften wir, ob der nächste Satz mit einem Großbuchstaben oder einer Zahl beginnt. So konnten wir schon relativ gut, die einzelnen Sätze erkennen. Allerdings gab es auch hier Ausnahmen, die diese Regeln erfüllten und trotzdem keine vollständigen Sätze waren. Hier einige Beispiele:

> *„Bisher betrachtet wurden vor allem sogenannte Blockkodes, d.h. ||Kodes, deren Elemente alle gleich lang waren.“*

> *„Diese Idee ist nicht neu, sondern lag schon lange vor Shannon dem Morsealphabet (S.F.B. ||Morse, 1791-1872) zugrunde.“*

> *„Einen ersten Ansatz zur Konstruktion eines solchen Baumes zur Kodierung von Zeichen mit bekannter Wahrscheinlichkeit lieferten Shannon und R.M.|| Fano fast zeitgleich in den späten 40ern:“*
> [Katz 2002]

Diese Beispielsätze aus einem Text über die Huffman-Kodierung würden fälschlicher Weise an den markierten Stellen in je zwei Sätze zerlegt werden. Auch bei der Erkennung von Sätzen in HTML-Texten gibt es dieses Problem. Hier entschieden wir uns dafür, dass sehr kurze Sätze (weniger als vier Worte) an den vorherigen Satz mit angegliedert werden. Außerdem wird bei allen Sätzen überprüft, ob das letzte Wort eine bekannte Abkürzung ist. Wenn diese Bedingung erfüllt ist, wird der Inhalt des folgenden Satzes an diesen Satz angehangen. Trotz dieser Versuche nur vollständige Sätze zu erkennen, gibt es hier ab und zu falsch erkannte Sätze. Da dieses Problem aber in sehr wenigen Sätzen auftrat, konnte es vernachlässigt werden.

Ein damit verwandtes Problem, was bewirkte, dass mehrere Sätze als ein einziger Satz erkannt wurden, rührte daher, dass bei einigen Texten die Formatierung fehlerhaft war. Das heißt, es fehlte nach einem Satzendzeichen das vorgeschriebene Leerzeichen. Da aber genau nach der Kombination aus Satzendzeichen und Leerzeichen für die Aufsplittung gesucht wird, konnten diese Sätze nicht richtig zerlegt werden. Dieses Problem haben wir nicht behoben, da dies einerseits schlecht zu realisieren ist, da sonst beispielsweise das Aufsplitten von Datumsangaben oder Dezimalzahlen abgefangen werden müsste. Andererseits war dies auch

nicht nötig, da eine solch fehlerhafte Formatierung nur in sehr wenigen Texten und auch dann nur bei sehr wenigen Sätzen aufgetreten ist.
Ein weiteres Problem stellten hier auch die Style- und Formatanweisungen in Form von Tags dar, welche auch als Sätze interpretiert wurden. Da diese Pseudosätze aber meist keine Eigenschaften relevanter Sätze aufweisen, werden sie auch meist nicht für die Zusammenfassung ausgewählt. Dies kann allerdings zu falschen Formatierungen in der späteren Zusammenfassung führen.
Was sich auch als etwas problematisch erwies, war das Fehlen eines Textkorpus, d.h. einer Sammlung von Texten mit dazugehörigen Zusammenfassungen, welche möglichst durch Satzextraktion entstanden sind. So mussten wir uns beispielsweise Hinweisphrasen oder so genannte Bonus- und Stigmaworte mühsam selbst erarbeiten, indem wir in verschiedenen Texten selbst die wichtigsten Sätze markierten und dann nach häufig vorkommenden Worten suchten. Dies war zeitaufwändig und die so entstandenen Wortlisten sind sicher nicht vollständig und korrekt.
Ein weiteres Hindernis bei der Implementierung war das Fehlen einer Funktion zur Stammformenreduktion. Solch eine Funktion wäre für den Luhn- und den Edmundson-Algorithmus von Nöten gewesen, da hier die Satzrelevanz ganz oder teilweise über die Worthäufigkeit berechnet wird und so Worte einer Wortfamilie zusammengezählt werden könnten. Aber auch meine Umsetzung ohne das Zusammenrechnen verwandter Worte brachte brauchbare Ergebnisse.
Auch die Erkennung von Eigennamen brachte Probleme mit sich. Wie bereits erwähnt, werden noch immer Wortgruppen irrtümlicher Weise als Eigennamen erkannt. Hier einige Beispiele:

> *„Wir haben aus den Grundlagen der Informationstheorie bereits eine wichtige Idee Shannons kennengelernt: Kontextunabhängige Wahrscheinlichkeiten lassen es zu, jedem Zeichen einen Informationsgehalt zuzuordnen als I(x)=-ld(p(x))(mit p(x) als Wahrscheinlichkeit des Zeichens x), gemessen in bit (basic indissoluble information unit).“*

> *„Nicht lange nach Shannon und Fano veröffentlichte D.A. Huffman 1952 ein Verfahren, das etwas bessere Kodes als Shannons und Fanos Verfahren produzierte, und das vor allem unter der Annahme einer Shannonschen Quelle*

nachweisbar einen präfixfreien Kode liefert, der optimal unter den präfixfreien Kodes ist in dem Sinne, daß er die kodierte Nachricht minimiert.“

„Konstruierten Shannon und Fano den Baum von oben nach unten, indem sie eine Liste immer weiter aufteilten, begann Huffman den Baum von unten aufwärts zu konstruieren.“
[Katz 2002]

Einige der Wortgruppen enthalten zwar auch Nachnamen von Personen, aber eigentlich ist dies zufälliger Natur. Es ist mir nicht gelungen, die Regeln für Eigennamen von Personen so zu formulieren, dass wirklich nur Eigennamen erkannt werden. Dies ist aber meiner Meinung nach auch nicht möglich, da beispielsweise bei der bloßen Erwähnung des Nachnamens einer Person selbst unter Nutzung einer Liste mit allen möglichen Vor- und Nachnamen wahrscheinlich nicht immer zweifelsfrei entschieden werden kann, ob es sich um ein einfaches Substantiv oder eine Person handelt. Besonders wenn der Nachname der Person auch als Substantiv eine Bedeutung hat, wie das zum Beispiel bei Helmut Kohl, Joschka Fischer oder Andrea Berg der Fall ist. Daher berücksichtigt mein Algorithmus nur Eigennamen, die aus mindestens zwei Worten bzw. aus Initialen und Worten bestehen. Da der überwiegende Teil der so erkannten Eigennamen wirklich Eigennamen oder Wortgruppen mit einem Eigennamen sind, konnte dieses Problem vernachlässigt werden. Eine mögliche Verbesserung könnte dadurch erreicht werden, dass ein erkannter Eigenname erst dann gewertet wird, wenn er mindestens zweimal im Text zu finden ist. So könnte man mit sehr hoher Wahrscheinlichkeit annehmen, dass es sich um einen „echten“ Eigennamen handelt. Außerdem ist dadurch auch gewährleistet, dass die benannte Person eine gewisse Relevanz aufweist, da sie mehrmals im Text erwähnt wird.

5.2. Probleme bei der Evaluierung

Auch bei der Evaluation der implementierten Algorithmen gab es einige Schwierigkeiten. So zeigte sich im Nachhinein, dass die Wahl des dritten Testtextes etwas unglücklich ausgefallen war. Hier wurden von den Teilnehmern der Nutzerbefragung eher Sätze markiert, deren Inhalt die Nutzer interessierte, die aber nicht einen allgemeinen Überblick über den Inhalt des Textes gaben. Ein weiteres Hindernis war die Länge der Texte. Um die Zusammenfassungsalgorithmen wirklich effizient testen zu können, wären längere Texte

sinnvoll gewesen. Dies war aber nicht möglich, da sonst die Nutzerbefragung um ein vielfaches länger gedauert hätte, was möglicherweise die freiwilligen Teilnehmer zur Nichtteilnahme veranlasst hätte. Ein eher kleines Problem war, dass einige Teilnehmer die Aufgabenstellung scheinbar nicht genau gelesen hatten und keine ganzen Sätze sondern nur Wortgruppen unterstrichen hatten. Dieses Problem konnte aber ganz einfach gelöst werden, indem jeder Satz, in dem mindestens eine Wortgruppe markiert war, als selektiert angesehen wurde.

Bei den Tests der verschiedenen Extraktionsalgorithmen bestand eine Schwierigkeit darin, möglichst sinnvolle und aussagekräftige Parameterkombinationen zu finden. Wir wählten die Parameter dann so, dass der Einfluss einer Komponente oder Methode möglichst gut sichtbar war. Daher wurde nur 0 oder 1 als Gewicht vergeben. Die so gefundenen Parametrisierungen sind sicher nicht die optimalsten, aber wir sind der Meinung, dass die Beeinflussung der verschiedenen Komponenten nur so erkennbar ist. Interessant wäre natürlich auch ein Vergleich mit der besten Parameterkombination gewesen. Diese zu ermitteln wäre allerdings zu aufwändig gewesen.

5.3. Optimierung unseres Algorithmus zur Verbesserung der Platzierung im Ranking

Wir sind der Meinung, dass unser Algorithmus im obigen Ranking den ersten Platz belegt hätte, wenn die Testtexte im HTML-Format vorgelegen hätten, da unser Algorithmus bei diesem Dateiformat viel mehr Komponenten eines Satzes (Links, Bilder nach dem Satz) zur Bewertung des Satzes berücksichtigt. Aber auch bei Texten im RTF-Format ist für unseren Algorithmus der erste Platz im obigen Ranking erreichbar, wenn einige Verbesserungen vorgenommen werden.

5.3.1. Verbesserung der Suchwort-Funktion

Eine Verbesserung würde die Bearbeitung der Suchwort-Funktion mit sich bringen. So könnte hier ein Thesaurus eingesetzt werden, der es ermöglicht, dass nicht nur nach den eingegebenen Suchworten gesucht wird, sondern auch nach verwandten Worten. Wird zum Beispiel „Renault“ als Suchwort eingegeben, könnte die Suchwortliste automatisch um Begriffe wie „Auto“, „PKW“ oder „Fahrzeug“ erweitert werden, welche aber etwas weniger Gewicht haben als der vom Nutzer eingegebene Begriff. So könnten bei Eingabe von „Vogelgrippe“ in die Suchwortliste verwandte Worte wie „Geflügelpest“, „Tierseuche“,

„Seuche“, „Virus“ oder „aviäre Influenza“ hinzugefügt werden. Sinnvoll wäre der Thesaurus auch bei der Suche nach den eingegebenen Suchworten. So könnte beispielsweise auch nach anderen Formen (Singular-Plural, andere Zeitformen) der eingegebenen Suchworte im Text gesucht werden.

5.3.2.Optimierung der Gewichte

Am einfachsten ließe sich die Ranking-Position verbessern, indem die Gewichte für die einzelnen Komponenten optimiert werden. Dies würde aber sehr viele Tests mit vielen Texten und unterschiedlichsten Gewichtskombinationen erfordern.
In einem von uns durchgeführten Versuch mit den drei Evaluationstexten konnten schon allein durch die Erhöhung einzelner Werte geringfügig bessere Recall- und Precisionwerte erreicht werden. So konnten beim Ayla-Text auch ohne die Angabe von Suchworten mit der Erhöhung der Gewichte des Positions- und des ÜberschriftenInhaltsWertes bei der Auswahl von 5 Sätzen Recallwerte von 57 % und Precisionwerte von 80% erreicht werden. Bei dem Text über die SPD-Klausur konnten ohne Angabe von Suchworten und mit einer Verminderung des Gewichtes der Eigennamen Recallwerte von 33% und Precisionwerte von 50% erreicht werden. Bei der Angabe von Suchworten, die besser auf die Nutzerinteressen abzielten („Familie“, „Kinder“, „Beruf“), konnten bei gleichen Gewichten sogar ein Recall von 50% und eine Precision von 75% gemessen werden.
Da diese Verbesserungen schon durch ein bisschen „Herumprobieren“ erzielt werden konnten, sehen wir in der Optimierung der Gewichte ein sehr großes Potential, um die Zusammenfassungen noch besser zu machen.

5.3.3. Anpassung der Schlüsselworte an das Textgenre oder das Textthema

Bei der Evaluierung hat sich gezeigt, dass die in der Datei hinterlegten Schlüsselwörter fast gar nicht in den Testtexten vorkamen. Dies lag daran, dass die Schlüsselworte eher für wissenschaftliche Texte sinnvoll gewesen wären.
Um wichtige Sätze besser zu erkennen, wären Schlüsselworte von großem Nutzen, welche passend zum Textgenre ausgewählt werden können. Allerdings ist dies unserer Meinung nach bei Zeitungstexten recht schwierig, da sowohl in den Überschriften als auch im laufenden Text durch die sehr unterschiedlichen Themen, welche Zeitungsartikel behandeln, kaum eindeutige Schlüsselworte gefunden werden können.

Als Lösung für dieses Problem und somit zur Verbesserung der Ranking-Position unseres Algorithmus würden wir vorschlagen, Schlüsselwortlisten dynamisch mit Hilfe des Internets passend zu jedem Text generieren zu lassen. So könnte man vielleicht die eingegebenen Suchworte und den Titel des Textes in eine Google-Anfrage eingeben und die 10 ersten Suchergebnisse auf Schlüsselworte untersuchen. Als Schlüsselworte würden Substantive betrachtet, die in einer bestimmten Häufigkeit auf diesen Seiten zu finden sind.

5.3.4. Berücksichtigung der Satzlänge

Bei der Analyse der von den Nutzern als relevant eingestuften Sätze fiel uns auf, dass ein Großteil dieser Sätze sehr lang war. Daraufhin untersuchten wir das Verhältnis zwischen Länge des Satzes und Relevanz näher und stellten fest, dass stets mindestens die Hälfte der extrahierten Sätze überdurchschnittlich lang war.

Text	Durchschnittliche Satzlänge	Verhältnis überdurchschnittl. langer, relevanter Sätze zur Gesamtanz. relev. Sätze
Text1 (Ayla)	12 Worte	$\frac{4}{7}$
Text2(Vogelgrippe Türkei)	21 Worte	$\frac{5}{10}$
Text3 (SPD Klausur)	18 Worte	$\frac{3}{6}$

Tabelle 27: Zusammenhang zwischen Satzlänge und Relevanz

Daraus schlussfolgerten wir, dass durch eine stärkere Berücksichtigung der Satzlänge, die Zusammenfassungen besser werden müssten und sich somit die Position im Ranking verbessern würde. Man sollte unseren Algorithmus dahingehend ändern, dass eine Satzlänge, die über dem Durchschnitt liegt, eine Gewichtserhöhung dieses Satzes nach sich zieht. Mit welchem Gewicht die Länge in die Berechnung des Gesamtgewichtes eingeht, müsste dann auch über die GUI steuerbar sein.
Bei HTML-Texten ist aber zu beachten, dass nur „richtige“ Worte, d.h. keine Styleanweisungen, zur Berechnung der Satzlänge herangezogen werden.

5.4. Erweiterungen

Das erstellte Zusammenfassungsprogramm bietet noch einige Möglichkeiten zur Erweiterung. Eine sinnvolle Erweiterung wäre zum Beispiel die Möglichkeit, dass Zusammenfassungen in

unterschiedlichen Formaten, nicht nur in HTML, erstellt werden können. Allerdings wären dann die als relevant bewerteten Links nicht mehr so sinnvoll.

5.4.1. Zusammenfassungen von Grafiken

Zurzeit wird hauptsächlich auf dem Gebiet der Automatischen Textzusammenfassung Forschung betrieben und die im Original enthaltenen Grafiken und Diagramme werden so wie sie sind mit in die Zusammenfassung übernommen. Es gibt aber auch schon Forschungsarbeiten, die sich mit dem Zusammenfassen von Grafiken und Diagrammen auseinandersetzen (vgl. [Futrelle 1999]). Eine sinnvolle Erweiterung unseres Zusammenfassungsprogramms wäre daher eine integrierte Funktion zur Zusammenfassung von grafischen Elementen des Textes. So könnte die Größe der Zusammenfassung reduziert werden und die Aussage der Grafik oder des Diagramms bliebe in knapperer, aber präziser Form erhalten. Das folgende Beispiel verdeutlicht die Zusammenfassung einer Grafik:

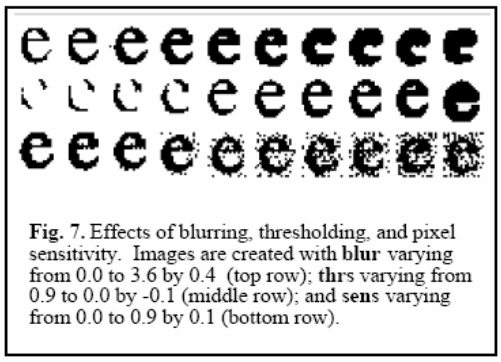


Fig. 7. Effects of blurring, thresholding, and pixel sensitivity. Images are created with **blur** varying from 0.0 to 3.6 by 0.4 (top row); **thr**s varying from 0.9 to 0.0 by -0.1 (middle row); and **sen**s varying from 0.0 to 0.9 by 0.1 (bottom row).

Abbildung 58: Originalgrafik [Futrelle 1999, S.2]

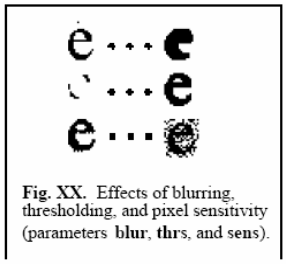


Fig. XX. Effects of blurring, thresholding, and pixel sensitivity (parameters **blur**, **thr**s, and **sen**s).

Abbildung 59: mögliche Zusammenfassung der Grafik [Futrelle 1999, S. 2]

5.4.2. Erstellen von Zusammenfassungen verschiedener Sprachen

Ein anderer Ansatzpunkt unser Zusammenfassungssystem zu erweitern, wäre die Möglichkeit, Texte in verschiedenen Sprachen zusammenzufassen. Dafür müssten die Cue-, Heading- und Nulldateien in der Sprache des Originals vorliegen. Des Weiteren müssten einige im Programm enthaltene Wortlisten, wie die der Konjunktionen, auch in Dateien ausgelagert werden, um so schnell durch die Datei in der entsprechenden Sprache ersetzt zu werden. Denkbar und auch sinnvoll wäre hier auch die Kombination mit einem Übersetzungsprogramm. So könnten beispielsweise von englischen oder spanischen Texten deutsche Zusammenfassungen angefertigt werden, welche dem Nutzer dann die Entscheidung erleichtern, ob der Originaltext für ihn relevant ist oder nicht.

5.4.3. Erstellen noch kompakterer Zusammenfassungen durch zusätzliche Satzreduktion

Um die Zusammenfassungen noch kürzer zu machen und so irrelevante Satzteile zu entfernen, könnte man unser Zusammenfassungssystem dahin gehend erweitern, dass nach der Auswahl der relevanten Sätze diese Sätze durch Satzreduktion verkürzt werden. Auf diesem Weg könnten uninteressante Neben- oder Relativsätze entfernt werden, so dass nur die Kernaussage des Satzes übrig bleibt. Allerdings muss das Reduktionsprogramm wissen, welche Satzteile entfernt werden können und welche für den Sinn des Satzes unverzichtbar sind. Jing (vgl. [Jing 2000]) nutzte dafür einen entsprechend großen Korpus mit Originalsätzen und den dazugehörigen reduzierten Sätzen um sein Programm zu trainieren (siehe Kapitel 2.4.2.). Dies ist natürlich sehr aufwendig und zeitintensiv.
Durch eine solche Erweiterung könnten eventuell sogar mehr Sätze, welche dann entsprechend kürzer sind, in die Zusammenfassung übernommen werden, ohne dass die Zusammenfassung dadurch die vorgegebene Größe überschreitet.

5.4.3. Erstellen von Multidokumenten-Zusammenfassungen

Eine letzte Erweiterung wäre die Erstellung einer Zusammenfassung von mehr als einem Originaldokument. Wenn mehrere Texte zu einem Thema vorliegen, kann man so die Quintessenz aller dieser Texte in einer Zusammenfassung wiedergeben. Wichtig dabei wäre, Redundanzen zu vermeiden, da bei mehreren Texten zu ein und demselben Thema sicher einige Fakten wiederholt werden. Bis auf die Vermeidung redundanter Informationen ist diese Erweiterung recht einfach umzusetzen. Es werden nun nicht nur die Satzgewichte für alle Sätze eines Textes berechnet, sondern für alle Sätze aller Texte. Dann wählt man die höchstgewichtigsten Sätze aus und überprüft sie auf redundante Informationen. Danach erhält man die relevantesten Sätze aller Texte, welche in geordneter Form die Zusammenfassung ergeben.

6. Anwendungsmöglichkeiten

Da Zusammenfassungen in so vielen Bereichen des täglichen Lebens zu finden sind, sind auch die Einsatzgebiete für Automatische Textzusammenfassungssysteme relativ vielseitig.

6.1. Kombination von Suchmaschinen und Zusammenfassungssystemen für die Webrecherche

Auf der Suche nach Informationen im WorldWideWeb wäre man ohne den Einsatz von Suchmaschinen ziemlich verloren. Allerdings erhält der Nutzer oft tausende bis Millionen von eventuell relevanten Seiten angeboten (siehe ovale durchgängige Markierung in Abbildung 60 und 61).

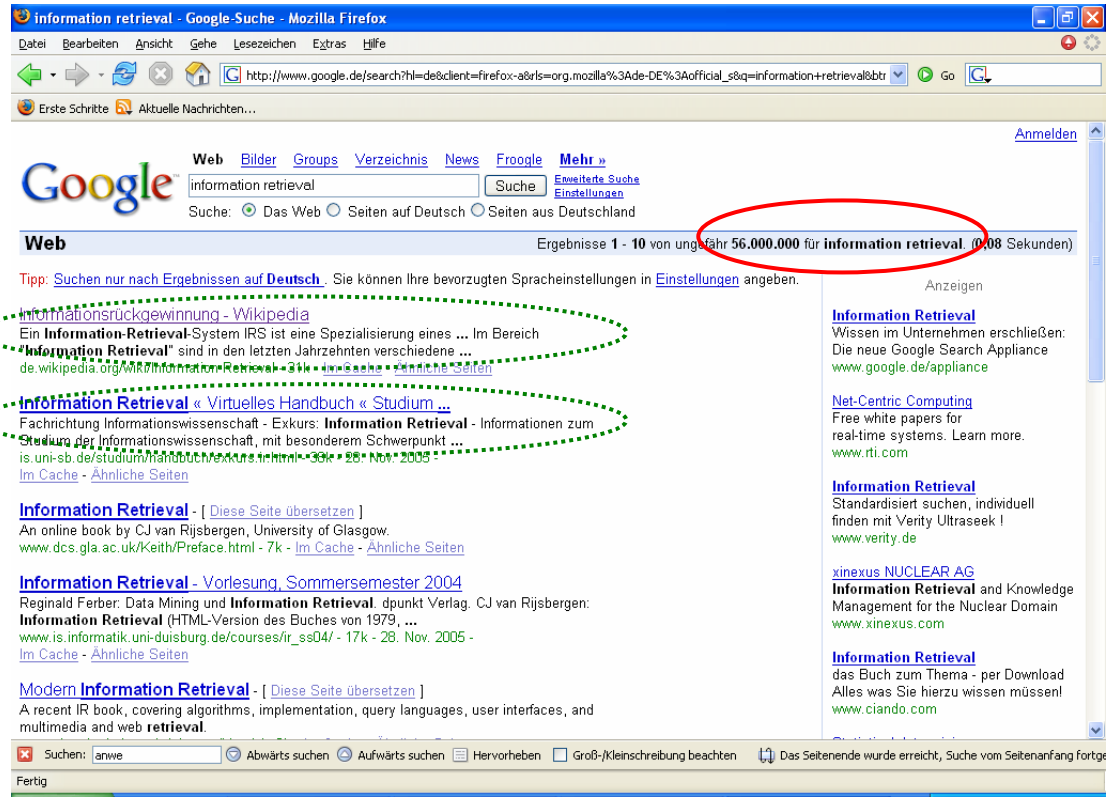


Abbildung 60: Auszug aus den Ergebnissen von Google zum Thema "information retrieval" am 29.11.05

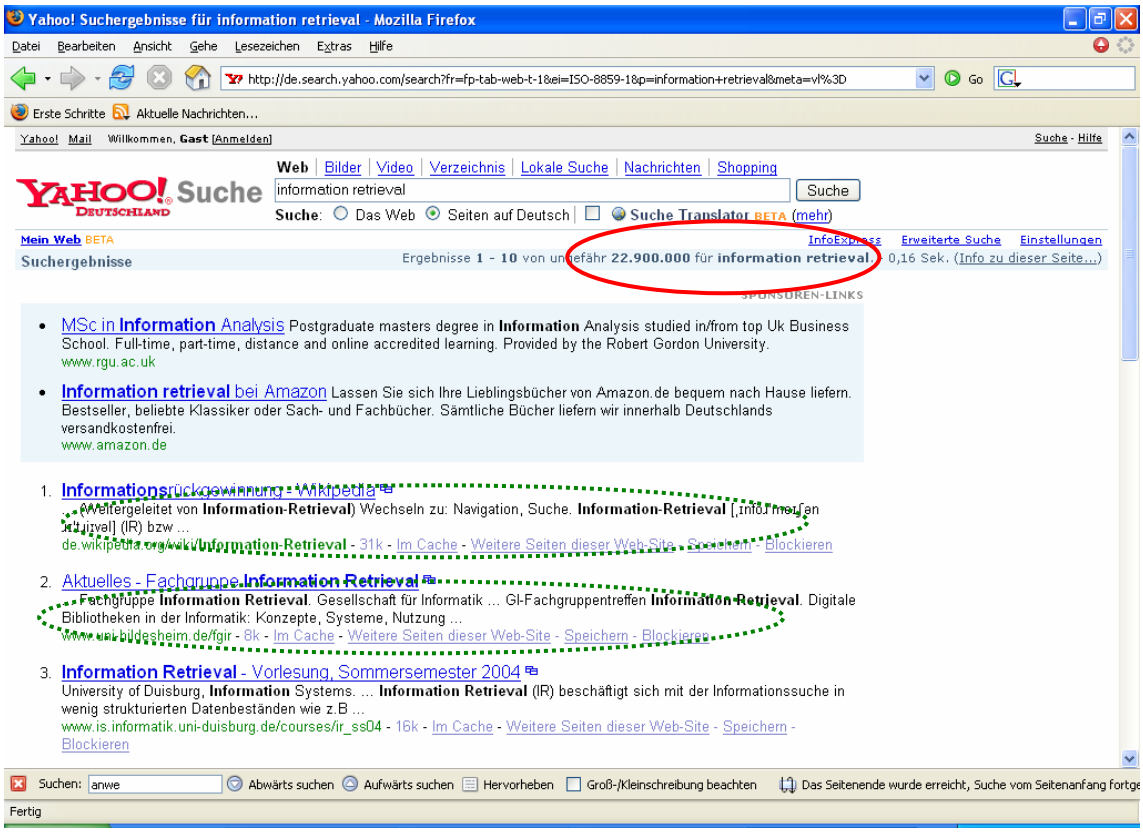


Abbildung 61: Auszug aus den Ergebnissen von Yahoo zum Thema "information retrieval" am 29.11.05

Auch die zu jeder Trefferseite angezeigte Überschrift der Seite sowie der Auszug aus den ausgegebenen Ergebnisseiten mit den fett hervorgehobenen Suchwörtern (siehe gestrichelte Markierung in Abbildung 60 und 61) bieten jedoch meistens keine hinreichenden Informationen über die Relevanz der Trefferseite. Dadurch ist der Nutzer genötigt, dem Link zur Trefferseite zu folgen. Gerade bei der Benutzung eines Modems oder bei Trefferseiten mit vielen Bildern kann das eine gewisse Zeit dauern. Enttäuschend ist danach die eventuelle Einsicht, dass die ausgewählte Seite irrelevant ist.

In diesem Fall könnte sich die Kombination von Suchmaschinen und Textzusammenfassungssystemen als sehr nützlich erweisen. So könnte unter der Überschrift der Seite eine sehr knappe (10 – 20 % des Originals oder 3 – 5 Sätze) nutzerorientierte Zusammenfassung präsentiert werden. Die Nutzerorientierung kann dabei durch Berücksichtigung der in der Suchanfrage enthaltenen Schlüsselworte erfolgen.

6.2. Einsatz von Zusammenfassungssystemen im eLearning

Für einen Lehrer oder Tutor, welcher eLearning nutzen möchte, bringt dies zu Beginn nicht die so oft propagierte Zeitersparnis mit sich. Am Anfang steht der Lehrer vor der Aufgabe, den Unterrichtsstoff in das eLearning-System einzutragen. Dies ist ein sehr zeitaufwändiger und fortlaufender Prozess, denn neben dem vielleicht neuen und unbekannten Programm zur Erstellung des eLearning-Kurses, welches erlernt werden muss, müssen auch die Inhalte recherchiert und später während des Kurses ständig auf Aktualität überprüft und eventuell geändert werden und das neben der Betreuung der Lernenden.

Für diese Lehrer, aber auch für Schüler, welche einen eLearning-Kurs zu einem bestimmten Thema nutzen möchten und kein passendes Angebot finden, kann die Kombination eines eLearning-Systems mit einem System für die Automatische Textzusammenfassung eine hilfreiche Alternative sein.

Denkbar wäre zum Beispiel, dass in das eLearning-System das gewünschte Themengebiet eingegeben wird, zu welchem ein eLearning-Kurs gesucht wird. Das eLearning-System könnte diese Anfrage an das mit dem Internetverbundene Zusammenfassungssystem weitergeben. Dieses würde eine bestimmte Anzahl an Texten zu dem gewünschten Thema mit Hilfe einer Suchmaschine aus dem Internet oder aus einer dafür angelegten Datenbank einer Universität holen und aus diesen Texten eine Zusammenfassung erstellen. Diese Zusammenfassung könnte dann als Inhalt in den eLearning-Kurs eingetragen werden. Der Tutor könnte dies als Grundgerüst für seinen eLearning-Kurs benutzen und diesen beliebig erweitern. Ein Schüler könnte sich mit diesem Kurs einen Überblick über das Thema verschaffen und durch die Verfolgung von enthaltenen Links sein Wissen vertiefen.

6.3. Anwendungsmöglichkeit im Information Retrieval

Beim Information Retrieval (deutsch: Informationswiedergewinnung oder Informationsbeschaffung) ist der Kernpunkt die Computerunterstützte inhaltsorientierte Suche. Ein Nutzer eines IR-Systems verfolgt ein bestimmtes Ziel oder eine bestimmte Aufgabe, wofür er Informationen benötigt. Diese Informationen können in Dokumenten, welche in dem IR-System gespeichert sind, enthalten sein. Um nun für seine Aufgabe relevante Dokumente aufzufinden, sendet der Nutzer eine Anfrage an das System. Das IR-System vergleicht dann diese Anfrage mit den im System eingestellten Dokumenten. Als Antwort auf seine Anfrage erhält der Nutzer dann eine Liste eventuell relevanter Texte. Diese Texte muss der Nutzer nun auf ihre Brauchbarkeit für seine Aufgabe hin untersuchen. Diesen

Schritt könnten Zusammenfassungssysteme vereinfachen. Anstatt des kompletten Originaltextes könnte der Nutzer im ersten Schritt eine allgemeine 10%ige Zusammenfassung (d.h. eine Zusammenfassung, die eine Länge von 10% des Originals aufweist) angezeigt bekommen. So kann er schneller entscheiden, ob dieser Text für ihn von Interesse ist oder nicht. Sollte der Nutzer über die Relevanz des Textes noch unschlüssig sein, wäre es denkbar, dass er im zweiten Schritt eine 20%ige oder 50%ige Zusammenfassung angeboten bekommt. Ist der Nutzer sicher, dass der Text für seine Aufgabe relevant ist, kann er sich im letzten Schritt den Originaltext präsentieren lassen und damit seine Aufgabe lösen.

Denkbar wäre auch, dass nicht eine allgemeine Zusammenfassung geboten wird, sondern eine auf die Nutzerinteressen ausgerichtete. Dies könnte über die in die Anfrage eingegebenen Schlüsselworte geschehen.

Durch die Nutzung von automatischen Textzusammenfassungssystemen im Information Retrieval können die Anwender so viel Zeit sparen, da sie nicht den kompletten Text lesen müssen, um dann zu dem Urteil zu kommen, dass er irrelevant ist.

Literaturverzeichnis

[Alonso i Alemany 2005]

Alonso i Alemany, L. (2005). Representing discourse for automatic text summarization via shallow NLP. Online im Internet: http://www.cs.famaf.unc.edu.ar/~laura/shallowdisc4summ/tesi_electronica.pdf (05-12-01).

[Amigó, Gonzala, Peinado, Penas & Verdejo 2004]

Amigó, E., Gonzalo, J., Peinado, V., Penas, A., Verdejo, F. (2004). An empirical study of information synthesis task. Online im Internet: http://acl.ldc.upenn.edu/acl2004/main/pdf/303_pdf_2-col.pdf (05-12-02).

[Barzilay & Elhadad 1997]

Barzilay, R., Elhadad, M. (1997). Using lexical chains for text summarization. In M. Maybury and I. Mani (Hrsg.), Advances in Automatic Text Summarization (S. 111 - 122), Cambridge: MIT Press.

[Edmundson 1969]

Edmundson, H. P. (1969). New methods in automatic extracting. Journal of the Association for Computing Machinery, 16(2), S. 274.

[Endres-Niggemeyer 2002]

Endres-Niggemeyer, B. (2002). Automatisches Textzusammenfassen. In H. Lobin & L. Lemnitzer (Hrsg.). Texttechnologie (S. 407-432). Tübingen: Stauffenburg.

[Euler 2001]

Euler, T.(2001). Informationsextraktion durch Zusammenfassung maschinell selektierter Textsegmente. Diplomarbeit. Dortmund: Fachbereich Informatik der Universität Dortmund.

[Firmin 1997]

T. Firmin Hand (1997): A Proposal for Task-based Evaluation of Text Summarization Systems. Online im Internet: http://acl.ldc.upenn.edu/W/W97/W97-0706.pdf.

[Futrelle 1999]

Futrelle, R. P. (1999). Summarization of Diagrams in Documents. In M. Maybury and I. Mani (Hrsg.), Advances in Automatic Text Summarization (S. 403 - 422), Cambridge: MIT Press.

[Guendogan & Schimpfky 2005]

Guendogan, A., Schimpfky, R. (2005). Text Summarization. Seminararbeit. Berlin: Fakultät für Informatik der Humboldt-Universität, S. 4

[Harnly 2005]

Harnly, A. (2005). Automation of Summary Evaluation by the Pyramid Method. Online im Internet: http://www1.cs.columbia.edu/~jaxin/nlpmeetings/2005-02-17-harnly.pdf (06-01-10).

[Hirao, Okumura, Fukushima & Nanba 2004]

Hirao, T., Okumura, M., Fukushima, T., Nanba, H.(2004). Text Summarization Challenge 3 – Text Summarization Evaluation at NTCIR Workshop 4. Online im Internet: http://research.nii.ac.jp/ntcir/workshop/OnlineProceedings4/TSC/NTCIR4-OV-TSC-HiraoT.pdf (05-10-13)

[Hovy & Marcu 1998]

Hovy, E., Marcu, D. (1998). COLING-ACL' 98 Pre-Conference Tutorial. Online im Internet: www.isi.edu/~marcu/coling-acl98-tutorial.html (05-10-13).

[Hovy & Lin 1999]

Hovy, E., Lin, C.Y. (1999). Automated Text Summarization in SUMMARIST. In M. Maybury and I. Mani (Hrsg.), Advances in Automatic Text Summarization (S. 81-94), Cambridge: MIT Press.

[Jing 2000]

Jing, H. (2000). Sentence Reduction for Automatic Text Summarization. In Proceedings of ANLP 2000. Online im Internet: http://citeseer.ist.psu.edu/541022.html (05-09-10)

[Jing, Barzilay, McKeown & Elhadad 1998]

Jing, H., Barzilay, R., McKeown, K., Elhadad, M. (1998). Summarization Evaluation Methods: Experiments and Analysis. Working Notes of the AAAI Spring Symposium on Intelligent Text Summarization, 60-68.

[Katz 2002]

Katz, B. (2002). Der Huffmancode. Online im Internet: http://www.irf.uka.de/seminare/rftk/huffman/ (05-11-03)

[Kluckhohn 2004]

Kluckhohn, K. (2004). Kleines Glossar zur "Einführung in die Sprachwissenschaft" .Online im Internet: http://www.uni-leipzig.de/~kluck/a1/glossar.htm (05-11-11)

[Kupiec 1995]

Kupiec, J., Pedersen, J., Chen, F. (1995). A Trainable Document Summarizer. Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 68-73.

[Larocca Neto, Santos, Kaestner & Freitas 2000]

Larocca Neto, J., Santos, A.D., Kaestner, A.A., Freitas, A.A. (2000). Generating Text Summaries through the Relative Importance of Topics. In M.C. Monard & J.S. Sichman (Hrsg.), Lecture Notes in Artificial Intelligence, No. 1952, (S. 300-309), Berlin: Springer-Verlag.

[Lin 2001]

Lin, C.Y. (2001).SEE Summary Evaluation Environment. User's Guide, Online im Internet: http://www.isi.edu/~cyl/SEE/SEEManual.pdf (05-06-25).

[Luhn 1958]

Luhn, H. P. (1958). The Automatic Creation of Literature Abstracts. IBM Journal of Research & Development, 2, 159-165.

[Mani & Bloedorn 1999]

Mani, I., Bloedorn, E. (1999). Summarizing similarities and differences among related documents Information Retrieval, 1, 35-67.

[Mani, Klein, House & Hirschman 2001]

Mani, I., Klein, G., House, D., Hirschman, L. (2001). SUMMAC: a text summarization evaluation. Online im Internet: http://acl.ldc.upenn.edu/E/E99/E99-1011.pdf

[Mani, Klein, House, Hirschmann, Firmin & Sundheim 1999]

Mani, I., Klein, G., House, D., Hirschmann, L., Firmin, T., Sundheim, B. (1999). The TIPSTER SUMMAC Text Summarization Evaluation.

Online im Internet: http://acl.ldc.upenn.edu/E/E99/E99-1011.pdf

[Mani, Klein, House, Hirschmann, Firmin & Sundheim 1998]

Mani, I., Klein, G., House, D., Hirschmann, L., Firmin, T., Sundheim, B. (1998). SUMMAC Final Report Part 2. Online im Internet: http://www.csi.uottawa.ca/tanka/ArtDB/summac-final-report-part2.ps.gz

[Mani, Klein, House, Hirschmann, Firmin & Sundheim 2002]

Mani, I., Klein, G., House, D., Hirschmann, L., Firmin, T., Sundheim, B. (2002). SUMMAC: a text summarization evaluation. Online im Internet: http://www.mitre.org/work/best_papers/best_papers_02/mani_summac/mani_summac.pdf

[Maybury 1995]

Maybury, M. (1995). Generating Summaries from Event Data. Information Processing and Management, 31(5), 735-751.

[McLellan, Tombros, Jose, Ounis & Whitehead 2001]

P. McLellan, A. Tombros, J. Jose, I. Ounis, M. Whitehead (2001). Evaluating Summarisation Technologies: A Task Oriented Approach. Online im Internet: http://www.dcs.qmul.ac.uk/~tassos/publications/nddl01.pdf.

[Morris, Kasper & Adams 1992]

Morris, A., Kasper, G., Adams, D. (1992). The Effects and Limitations of Automatic Text Condensing on Reading Comprehension Performance. In M. Maybury and I. Mani (Hrsg.), Advances in Automatic Text Summarization (S. 305-323), Cambridge: MIT Press.

[Nenkova & Passonneau 2004]

Nenkova, A., Passonneau, R. (2004). Evaluating content selection in summarization: the pyramid method. Online im Internet: http://www1.cs.columbia.edu/~ani/papers/pyramid.pdf (06-01-30).

[Paice & Jones 1993]

Paice, C. D., Jones, P. A. (1993). The Identification of Important Concepts in Highly Structured Technical Papers. Online im Internet: http://www.ics.uci.edu/~pratt/courses/papers/TextMining/paice-SIGIR.pdf (05-09-21)

[Radev & Tam 2003]

Radev, D.R., Tam, D. (2003). Summarization Evaluation using Relative Utility. Online im Internet: http://tangra.si.umich.edu/~radev/papers/p508-radev.pdf (06-01-10).

[Saggion & Lapalme 2000]

Saggion, H., Lapalme, G. (2000). Concept Identification and Presentation in the Context of Technical Text Summarization. Online im Internet: http://acl.ldc.upenn.edu/W/W00/W00-0401.pdf (05-12-10)

[Tombros & Sanderson 1998]

Tombros, A., Sanderson, M. (1998). Advantages of query biased summaries in information retrieval. Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2-10.

[van Halteren & Teufel 2004]

van Halteren, H. S. Teufel (2004). Evaluating information content by factoid analysis: human annotation and stability. Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, 419-426.

[Wyllys 1967]

Wyllys, R. E.(1967). Extracting and Abstracting by Computer. In Borko, H. (Hrsg), Automated Language Processing (S. 127-179), New York, NY: John Wiley and Sons.

Anhang 1 – DUC Evaluationsprotokoll

Procedure for human comparison of model (reference) and peer (system-generated and other) abstracts

- For each document set (in randomized order):
    - For each summary type (single-document or multi-document summary):
        - For each peer summary (in randomized order) - composed of peer units (PUs), which will be sentences:
            - If the peer target size is greater than 10, the evaluator reads the peer summary and then makes overall judgments as to the peer summary's quality, independent of the model. Questions 1-5 are within-sentence judgments; the rest are within- or across-sentence judgments. The answers are chosen in every case from the following set of 4 ordered categories: {0, 1-5, 6-10, more than 10}

1. About how many gross capitalization errors are there?

```
    Examples:
    - [new sentence] the new drugs proved
beneficial.
    - PEOPLE WERE AMAZED WHEN THEY LEARNED THE
DETAILS.
```

2. About how many sentences have incorrect word order?

```
    Examples:
    - John before Mary the park visited
```

3. About how many times does the subject fail to agree in number with the verb?

```
    Examples:
    - The student see a teacher with a telescope
    - It was clear, that the reporters agrees
with the idea.
```

4. About how many of the sentences are missing important components (e.g. the subject, main verb, direct object, modifier) - causing the sentence to be ungrammatical, unclear, or misleading?

```
    Examples:
```

```
    - The agreement, signed in London yesterday,
.
    - Mr. Smith to Washington where he met the
senator.
    - The exchange rate is %.
    - Stewart, the builder.
```

5. About many times are unrelated fragments joined into one sentence?

```
    Examples:
    - They run a refinery; two apples would be
enough.
```

6. About how many times are articles (a, an, the) missing or used incorrectly?

```
    Examples:
    - Men saw woman with the telescope
    - He picked up a book.  A book looked
interesting.
    - El Paso owns and operates refinery.
```

7. About how many pronouns are there whose antecedents are incorrect, unclear, missing, or come only later?

```
    Examples:
    - [opening sentence] Their agreement was
signed in Oslo in 1933.
    - Many Presidents were targets of assassins.
Pres. Reagan was
          wounded.  He was shot in the Ford's
Theater in 1865.
```

8. For about how many nouns is it impossible to determine clearly who or what they refer to?

```
    Examples:
    - The company agreed to negotiate.  [Which
company?]
```

9. About how times should a noun or noun phrase have been replaced with a pronoun?

```
    Examples:
            - Mr. John Smith went to DC.  Mr.
John Smith saw the senator.
```

10. About how many dangling conjunctions are there ("and", "however"...)?

```
    Examples:
    - [opening sentence]  However, they came to a
good agreement.
```

11. About many instances of unnecessarily repeated information are there?

```
    Examples:

    - Yesterday's estimate does not include any
projection for claims in
    Louisiana, which was also affected by the
storm, although less
    severely than Florida.  But on the Florida
losses alone, Hurricane
    Andrew becomes the most costly insured
catastrophe in the US.
    Louisiana was also affected by the storm.
With Florida's Hurricane
    Andrew losses added in, the total rises to
Dollars 11.2bn. This
    does not include claims in Louisiana.
```

12. About how many sentences strike you as being in the wrong place because they indicate a strange time sequence, suggest a wrong cause-effect relationship, or just don't fit in topically with neighboring sentences?

- View the model summary - composed of model units (MUs), which are human-corrected chunks of a type to be determined
- Evaluator steps through the MUs. For each MU s/he:
  - marks any/all PU(s) sharing content with the current MU
  - indicates whether the marked PUs, taken together, express about 0%, 20%, 40%, 60%, 80%, or 100% of the content in the current MU.
- Evaluator reviews unmarked PUs and indicates once for the entire peer summary that:
  - About 0%, 20%, 40%, 60%, 80%, or 100% of the unmarked PUs are related but needn't be included in the model summary

- (Evaluators will be allowed to review and revise earlier peer summary judgments before moving to the next document set - to mitigate learning effects.)

Anhang 2 – TIPSTER Thema 258

themenbezogene Fragen und ein Teil der relevanten Quelle mit Antwortmarkierungen

Title : Computer Security
Description : Identify instances of illegal entry into sensitive computer networks by nonauthorized personnel.
Narrative : Illegal entry into sensitive computer networks is a serious and potentially menacing problem. Both 'hackers' and foreign agents have been known to acquire unauthorized entry into various networks. Items relative to this subject would include but not be limited to instances of illegally entering networks containing information of a sensitive nature to specific countries, such as defense or technology information, international banking, etc. Items of a personal nature (e.g. credit card fraud, changing of college test scores) should not be considered relevant.
Questions
1)Who is the known or suspected hacker accessing a sensitive computer or computer network?
2) How is the hacking accomplished or putatively achieved?
3) Who is the apparent target of the hacker?
4) What did the hacker accomplish once the violation occurred?
What was the purpose in performing the violation?
5) What is the time period over which the breakins were occurring?
Annotated Source Fragment
As a federal grand jury decides whether he should be prosecuted, <QDa graduate student</Ql> linked to a "virus" that disrupted computers nationwide <q5>last month</Q5>has been teaching his lawyer about the technical subject and turning down offers for his life story. .... No charges have been filed against <Ql>Morris</Ql>, who reportedly told friends that he designed the virus that temporarily clogged about <Q3>6,000 university and military computers</Q3> <Q2>linked to the Pentagon's Arpanet network</Q2>. . . . . .
[Mani, Klein, House, Hirschmann, Firmin & Sundheim 2002, S.61]

Anhang 3 – Erster Evaluierungstext

Lebenslange Haft für Ayla-Mörder

Der Mörder der sechsjährigen Ayla ist zu lebenslanger Haft verurteilt worden. Die Zwickauer Richter stellten zudem die besondere Schwere der Schuld fest. Damit ist eine vorzeitige Freilassung des Täters nach 15 Jahren Haft ausgeschlossen.
Der Verurteilte, ein 37 Jahre alter Mann, war wegen Mordes und sexuellen Missbrauchs vorbestraft. Er hatte gestanden, Ayla im Mai vergangenen Jahres auf dem Weg zur Schule entführt, sexuell misshandelt und getötet zu haben. Mit dem Urteil folgte das Gericht weitgehend dem Antrag der Staatsanwaltschaft.

Sicherheitsverwahrung wurde nicht verhängt

Der Forderung nach Sicherheitsverwahrung wurde allerdings nicht entsprochen. Die Staatsanwaltschaft hat die anschließende Sicherungsverwahrung für den Angeklagten beantragt. Die Verteidigung sah dazu keine Notwendigkeit. Während des Prozesses waren zwei Gutachter zur Schulfähigkeit des Angeklagten gehört worden. Sie kamen jedoch zu unterschiedlichen Urteilen.

Die Polizei in Zwickau suchte per Fahndungsblatt nach der kleinen Ayla.

Angeklagter legte Teilgeständnis ab

Zum Prozessauftakt am 17. November 2005 hatte der 37-jährige Mann ein Teilgeständnis abgelegt. Der arbeitslose Fliesenleger aus Reinsdorf bei Zwickau räumte ein, Ayla im Mai 2005 entführt, sexuell missbraucht und getötet zu haben. In seinem letzten Wort sagte der Angeklagte, er könne sich die Tat bis heute nicht erklären. Es tue ihm leid und er wisse, dass er das Unrecht nicht wieder gut machen könne.

Angeklagter saß bereits wegen Missbrauch im Gefängnis

Ayla war im Mai 2005 auf dem Weg zur Schule in ein Auto gezerrt, sexuell missbraucht und ermordet worden. Nach rund zwölf Stunden waren die Ermittler dem Angeklagten auf die Spur gekommen. Augenzeugen konnten sein Auto mit Zwickauer Kennzeichen beschreiben. Der 37-jährige Mann ist bereits einschlägig vorbestraft. Als 17-Jähriger hatte er eine ältere Frau auf der Straße erstochen. Dafür wurde er vom Bezirksgericht Chemnitz zu 15 Jahren Gefängnis verurteilt. 1997 musste er wegen sexuellen Missbrauchs von zwei Kindern für zwei Jahre ins Gefängnis. Der aus der Nähe von Zwickau stammende Mann ist selbst Vater zweier Kinder.

Anhang 4 – Zweiter Evaluierungstext

Tödliches Spiel mit Hühnerblut

Armut und Unwissenheit sind mit schuld an Ausbreitung der Vogelgrippe in der Türkei

Die Vogelgrippe hat ihre ersten Opfer außerhalb von Südostasien gefordert. Im Osten der Türkei starben ein 14-Jähriger und seine ein Jahr ältere Schwester an der Krankheit.

Von Susanne Güsten

Dogubeyazit. Erkältet seien die Kinder, sagte der diensthabende Arzt und schickte die Familie Kocygit nach Hause. Zwei Wochen ist das her. Nun sind Mehmet und Fatma tot, ihre Schwester Hülya schwebt in Lebensgefahr, der kleine Ali ist schwer krank – und die ganze Türkei ist entsetzt: Vogelgrippe lautet der wahre Befund, der für den 14-jährigen Mehmet und die 15-jährige Fatma zu spät kam. Der Junge starb am Sonntag, das Mädchen gestern, ihre elfjährige Schwester ringt auf der Intensivstation mit dem Tod.

Seuchen-Alarm seit Oktober

In Dogubeyazit am östlichsten Rand der Türkei spielt sich das Drama ab, bei dem erstmals Menschen außerhalb von Südostasien an der Vogelgrippe starben. Die Seuche bewegt sich auf Europa zu, daran besteht kein Zweifel – doch übertragen lässt sich der Fall auf europäische Verhältnisse nicht. Armut, Unwissenheit und Inkompetenz waren mit daran schuld, dass die Kinder sterben mussten.

Vogelgrippe-Alarm herrscht in der Türkei schon, seit im Oktober das Virus bei Truthähnen in Kiziksa im Westen des Landes festgestellt wurde. Bis in den unterentwickelten und verarmten Osten, wo längst nicht alle Menschen lesen können, sprach sich das nicht herum. Bei der Familie Kocyigit – wie bei vielen anderen Kleinbauern – wurden die Hühner weiter in Haus und Hof gehalten und in der Küche geschlachtet. Die inzwischen verstorbenen Geschwister spielten sogar mit den abgehackten Hühnerköpfen und warfen sie sich gegenseitig zu. Dass sie sich durch das Hühnerblut an ihren Händen mit dem Virus infizierten, gilt als wahrscheinlich.

Diagnose kommt zu spät

Dem Arzt an der Gesundheitsstation in Dogubeyazit kam die Vogelgrippe aber nicht gleich in den Sinn, als Zeki Kocyigit seine vier erkrankten Kinder vor zwei Wochen dort vorstellte. Erst drei Tage später, als der besorgte Vater erneut mit seinen fiebernden Kindern in der Klinik auftauchte, kamen die Mediziner ins Grübeln. „Ob die Kinder sich wohl an den Hühnern vergiftet haben könnten, die wir neulich geschlachtet haben?“, dachte der Vater während der Untersuchung laut nach – und da fiel beim Arzt der Groschen. Die Kinder wurden ins Universitätskrankenhaus der nächsten Großstadt Van eingewiesen und auf Vogelgrippe getestet. Bevor die ersten Ergebnisse aus dem Labor zurückkamen, war Mehmet schon tot.

„Negativ“, lautete der Laborbefund dennoch. „Der Junge hatte keine Vogelgrippe“, erklärte das Gesundheitsministerium, schickt vorsichtshalber aber weitere Proben an spezialisierte Labors in Ankara und Istanbul. Weil die Proben „wegen schlechter Witterung“ nicht auf dem Luftweg, sondern auf der Straße quer durch das große Land nach Istanbul gebracht wurde, verstrichen weitere zwei Tage, bis das dortige Labor am späten Mittwochabend seinen Befund abgeben konnte: Die Kinder hatten Vogelgrippe. Für Fatma Kocyigit kam die Diagnose zu spät. Sie starb acht Stunden nach Bekanntgabe des Befundes.

Selbst wenn die Diagnose früher gestellt worden wäre, hätte das Mädchen womöglich nicht gerettet werden können, denn in Van war das Vogelgrippe-Medikament Tamiflu nicht vorrätig. Zwei Tage mussten die Ärzte dort warten, bis das Medikament aus dem Westen der Türkei gebracht wurde. Erst am Montag trafen die Päckchen dort ein. Diese medizinische Unterversorgung ist im Osten der Türkei chronisch. Erst vor einem Jahr war dort kein Tollwut-Serum vorrätig, als Wölfe eine Kleinstadt überfielen und 15 Menschen zerfleischten. Und immer wieder sterben dort Kinder an Skorpion-Stichen, weil die örtlichen Kliniken das Gegengift nicht vorrätig haben.

Während die Ärzte in Van um das Leben der kleinen Hülya Kocyigit kämpften, wurden immer mehr Kinder mit Verdacht auf Vogelgrippe in das Krankenhaus eingeliefert und teils auf der Intensivstation behandelt. Die Kinderstation der Uniklinik wurde von allen anderen Patienten geräumt, um sie aufzunehmen; fünf Menschen wurden in die nächste Großstadt Erzurum verlegt, weil die Kapazitäten in Van erschöpft waren. Unter den neuen Verdachtsfällen waren allein acht Kinder einer weiteren Familie, die ebenfalls aus Dogubeyazit stammt und Hühner hält. Das Problem wurzele in der Armut von Ostanatolien, wo viele Menschen sich Hühner für den Eigenbedarf im Haus halten, sagte der türkische

Landwirtschaftsminister Mehdi Eker. Ein ernstes Problem ist das tatsächlich. Aber noch keine Gefahr für Europa.

Anhang 5 – Dritter Evaluierungstext

Klausur in Mainz

Die SPD hat ein Herz für Familien

Die SPD will sich künftig stärker in der Familienpolitik profilieren. So sollen 230.000 zusätzliche Betreuungsplätze für Kinder unter drei Jahren geschaffen werden. Die Sozialdemokraten wollen sich damit stärker von der Union abgrenzen.

Zum Abschluss ihrer zweitägigen Klausur in Mainz will die SPD-Spitze Leitlinien für den künftigen Kurs verabschieden. In einem Programm mit dem Titel "Wir sichern Deutschlands Zukunft" spricht sich die Parteiführung für Reformen nach skandinavischem Vorbild aus. Ähnlich wie in nordeuropäischen Ländern müsse auch in Deutschland weitaus stärker als bisher wirtschaftliche Dynamik und soziale Gerechtigkeit miteinander verknüpft werden, heißt es in der Beschlussvorlage des SPD-Vorstands.
Die Sozialdemokraten plädieren insbesondere für neue Wege zur Förderung von Familien. Notwendig sei eine zielgenaue Politik, die es jungen Männern und Frauen erleichtere, Kinderwünsche zu erfüllen, ohne ihre berufliche Perspektive zu gefährden. Mit der Tagung in Mainz soll auch die Debatte um das neue Grundsatzprogramm eröffnet werden, das im Herbst nächsten Jahres auf einem Parteitag verabschiedet wird.

Anspruch auf Kindergartenplatz

Im Entwurf für den Vorstandsbeschluss dazu setzt sich die SPD für die Schaffung von 230.000 zusätzlichen Betreuungsplätzen für Kinder unter drei Jahren ein. Zudem solle es einen Anspruch auf einen Kindergartenplatz ab dem zweiten Lebensjahr geben. Zumindest im letzten Jahr vor der Schule sollten Kinder auf jeden Fall einen Kindergarten besuchen, heißt es weiter. Dazu sollte das letzte Kindergartenjahr möglichst beitragsfrei sein. Das Engagement der SPD für die Familien soll die zentrale Botschaft der zweitägigen Beratung sein.
Damit wollen sich die Sozialdemokraten auch in Abgrenzung zum Koalitionspartner Union profilieren. In der SPD gibt es die Sorge, sie könne in der großen Koalition an eigenständiger Erkennbarkeit verlieren. Vor allem Vertreter der SPD-Linken fordern daher, die Partei solle sich bei der Klausur im Streit über die Gesundheitsreform und die staatlichen Lohnzuschüsse

klar von der Union abgrenzen. Bei der Klausur will die SPD-Spitze auch die Vorbereitungen für die fünf Landtagswahlen in diesem Jahr und für ein neues Grundsatzprogramm treffen.

Debatte um gebührenfreien Kitas

Unterdessen ist der Vorstoß von Bundesfamilienministerin Ursula von der Leyen (CDU) zur Abschaffung der Gebühren für Kindertagesstätten in den Ländern auf geteiltes Echo gestoßen. "Es ist gut, dass der Bund sich auch Gedanken um dieses Thema macht. Wir brauchen aber keinen Wettbewerb von wohlklingenden Vorschlägen, sondern einen Wettbewerb an realistischen, umsetzbaren Perspektiven", sagte der niedersächsische Kultusminister Bernd Busemann (CDU) der "Neuen Presse" in Hannover.

Der niedersächsische SPD-Fraktionschef Wolfgang Jüttner schlug vor, die Koalition solle auf die verbesserte steuerliche Absetzbarkeit von Kinderbetreuungskosten verzichten und die frei werdenden Mittel für gebührenfreie Kindergärten verwenden. "Wir sollten darüber nachdenken, statt steuerlicher Begünstigung in die Daseinsvorsorge zu investieren", sagte Jüttner, der "Berliner Zeitung". Er lehnte es zudem ab, die Kosten den Gemeinden aufzubürden: "Wir dürfen die Kommunen hier nicht alleine lassen."

DPA/AP/Reuters

Anhang 6 – Referenzzusammenfassung des zweiten Textes

Armut und Unwissenheit sind mit schuld an Ausbreitung der Vogelgrippe in der Türkei

Die Vogelgrippe hat ihre ersten Opfer außerhalb von Südostasien gefordert. Im Osten der Türkei starben ein 14-Jähriger und seine ein Jahr ältere Schwester an der Krankheit.

Nun sind Mehmet und Fatma tot, ihre Schwester Hülya schwebt in Lebensgefahr, der kleine Ali ist schwer krank – und die ganze Türkei ist entsetzt: Vogelgrippe lautet der wahre Befund, der für den 14-jährigen Mehmet und die 15-jährige Fatma zu spät kam.

Seuchen-Alarm seit Oktober

In Dogubeyazit am östlichsten Rand der Türkei spielt sich das Drama ab, bei dem erstmals Menschen außerhalb von Südostasien an der Vogelgrippe starben. Die Seuche bewegt sich auf Europa zu, daran besteht kein Zweifel – doch übertragen lässt sich der Fall auf europäische Verhältnisse nicht. Armut, Unwissenheit und Inkompetenz waren mit daran schuld, dass die Kinder sterben mussten.

Diagnose kommt zu spät

Weil die Proben „wegen schlechter Witterung“ nicht auf dem Luftweg, sondern auf der Straße quer durch das große Land nach Istanbul gebracht wurde, verstrichen weitere zwei Tage, bis das dortige Labor am späten Mittwochabend seinen Befund abgeben konnte: Die Kinder hatten Vogelgrippe. Selbst wenn die Diagnose früher gestellt worden wäre, hätte das Mädchen womöglich nicht gerettet werden können, denn in Van war das Vogelgrippe-Medikament Tamiflu nicht vorrätig. Diese medizinische Unterversorgung ist im Osten der Türkei chronisch.

Aber noch keine Gefahr für Europa.

Anhang 7 – Referenzzusammenfassung des dritten Textes

Klausur in Mainz

Die SPD hat ein Herz für Familien

In einem Programm mit dem Titel "Wir sichern Deutschlands Zukunft" spricht sich die Parteiführung für Reformen nach skandinavischem Vorbild aus. Die Sozialdemokraten plädieren insbesondere für neue Wege zur Förderung von Familien. Notwendig sei eine zielgenaue Politik, die es jungen Männern und Frauen erleichtere, Kinderwünsche zu erfüllen, ohne ihre berufliche Perspektive zu gefährden.

Anspruch auf Kindergartenplatz

Im Entwurf für den Vorstandsbeschluss dazu setzt sich die SPD für die Schaffung von 230.000 zusätzlichen Betreuungsplätzen für Kinder unter drei Jahren ein. Zudem solle es einen Anspruch auf einen Kindergartenplatz ab dem zweiten Lebensjahr geben. Dazu sollte das letzte Kindergartenjahr möglichst beitragsfrei sein.

Debatte um gebührenfreien Kitas